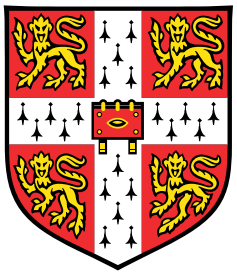
Multigrid methods for complex engineering geometries and unstructured meshes



Yuxuan Chen

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

St Edmund's College June 2020

Declaration

This thesis is the result of my own work and includes nothing which is the outcome of work done in collaboration except as declared in the Preface and specified in the text. It is not substantially the same as any that I have submitted, or, is being concurrently submitted for a degree or diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Preface and specified in the text. I further state that no substantial part of my thesis has already been submitted, or, is being concurrently submitted for any such degree, diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Preface and specified in the text. It does not exceed the prescribed word limit for the relevant Degree Committee.

Yuxuan Chen
June 2020

Abstract

Thesis title: Multigrid methods for complex engineering geometries and unstructured meshes
Author: Yuxuan Chen
Department of engineering

The convergence of standard multigrid methods decays significantly if locally poor quality cells are present, and it is found that the poor convergence is due to the local failure of the smoothing property. The high frequency error localised in regions of low quality cells is not eliminated by standard multigrid smoothers and persists through multigrid cycles. We propose a global–local combined smoother for the geometric multigrid to deal with engineering meshes with a small number of poor quality cells, which includes two steps: a global smoother on the whole domain, followed by a local correction on the subdomains with low quality cells. The high frequency error remaining in the low quality regions can be damped out completely by the local correction.

The idea is extended to the algebraic multigrid (AMG), including both classical AMG and smoothed aggregation AMG. It is suggested that the high frequency error produced by the smoother propagates outward the low quality region on the fine grid to the neighbouring areas on the coarse grid. An algorithm to track low quality regions on the abstract coarse grid of AMG has been developed based on the information transfer between grid levels via the transfer operators. With the local correction applied on low quality regions tracked on the abstract coarse grid, the high frequency error due to low grid quality can be removed.

In the smoothed aggregation AMG, the construction of the smoothed prolongation operator depends on the spectral radius of the system. However, regions of low quality cells in a mesh increase the largest eigenvalue of the linear system. We propose a shifted largest eigenvalue strategy to approximate a reasonable spectral radius to construct the smoothed prolongation.

Two and three dimensional numerical experiments, from illustrate to complicated, are demonstrated to validate the proposed smoother. Elliptic type PDEs, including Poisson and elasticity problems, are solved. For each example, the performance of multigrid on a high

quality mesh is also presented as a reference case, and it is shown that the poor convergence of multigrid for low quality meshes can be recovered to the reference case by the proposed smoother. A realistic thermomechanical simulation of turbomachinery problem has also been successfully solved.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Simulations of engineering problems commonly involve computing approximate solutions to partial differential equations (PDE). A widely used technique to solve PDEs in engineering is the finite element method, in which there is a step to solve a linear system. Modelling modern real–world engineering problems produces quite large and complicated linear systems, which require better treatments to solve. Therefore, it is imperative to develop methods to solve linear systems that are robust for complicated engineering applications.

Direct methods give the exact solution to a linear system (in the absence of round–off errors), but are prohibitive for larger problems due to the high computational cost. Iterative methods compute approximate solutions, which can sometimes be computationally cheap. Many advanced iterative solvers, such as multigrid and domain decomposition, have been developed to deal with different kinds of complex engineering problems. In particular, multigrid methods [27, 54, 108] have the potential to be optimal solvers, which means the amount of computational work involved in the method is proportional to the size of the linear system, i.e., $O(n)$ algorithm complexity, where n is the number of unknowns. Moreover, multigrid methods can lend themselves to efficient parallel implementation, which is hugely appealing in making possible high fidelity simulations of complex engineering components, and, even more attractive for computation of engineering at a system level.

Simulations of engineering applications, such as turbomachinery problems, are invariably performed on unstructured grids. The cell quality in grids is an important factor that affects the performance of linear solvers. Meshes with low quality cells are not uncommon in engineering problems since generation of meshes for complicated geometries in which all cells are of high quality can be difficult, especially for the case when representing geometrically complicated shapes with modest cell counts. Ideally, low quality meshes can be fixed at generation stage. However, even when possible, it may be very time consuming to produce a mesh with no regions of sub-optimal quality.

A natural question to raise is what the performance of iterative methods for poor quality meshes looks like, if it is not good, whether there is a way to formulate the iterative method to make it robust with respect to low cell quality. It has been recognised that the cell quality of mesh grid has a significant impact on the performance of iterative solvers [102, 45, 65]. The condition number of finite element systems increases with low quality meshes [35, 36]. The performance of multigrid also suffers from the use of low quality meshes [93, 44], which leads to our research in this thesis: we aim to understand why the performance of multigrid methods degrades with poor quality meshes, and to develop techniques to overcome poor solver performance in the presence of a small numbers of low quality cells.

In this chapter, we give a brief review of direct and iterative methods in section 1.1 including domain decomposition and multigrid. Some comments on the mesh quality are given in section 1.2. A simple example is presented in section 1.3 to illustrate the slow convergence of iterative methods on low quality meshes. With the specific motivation and goal, we give an overview of this thesis in section 1.4, including a summary of key research questions.

1.1 Methods for solving linear system

The finite element method is widely used in engineering simulations, and it includes a step to solve the linear system

$$Au = b, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is a matrix, $b \in \mathbb{R}^n$ is the right-hand side (RHS), and u is the unknown vector to solve. In this thesis, we focus on solving elliptic type PDEs including Poisson and elasticity problems, and symmetric positive-definite (SPD) matrices need to be solved. There are many types of solvers to solve such SPD matrix system, while the speed and accuracy vary a lot depending on the problem. Choosing a suitable solver to solve the matrix system is essential to the overall performance.

Algorithmic complexity plays an important role in numerical solvers, which stands for the computational cost with respect to problem size. An optimal solver has an $O(n)$ algorithm complexity, where n is the size of problem, meaning that the computational cost increases linearly with the problem size.

1.1.1 Direct methods

A direct method is the method to solve the linear system for the exact solution (in the absence of round–off errors). Direct methods are usually based on factorisation of the target matrix.

Common direct methods include the Gaussian elimination, the *LU* factorisation, and the Cholesky factorization [107]. The main drawback of using direct solvers is that they are usually time costly for large problems. For example, standard *LU* factorisation takes $O(n^3)$ algorithm complexity for a dense matrix. And a sparse *LU* factorisation can have $O(n^2)$ complexity for a three-dimensional finite element problem [47, 90], which $O(n^2)$ is better than $O(n^3)$, but it still scales powerly for large n. Even if a high-level parallel computing is applied, the high time cost of direct solvers with large size problems can never be conquered. If the problem size is in $O(10^9)$, then the time cost of direct solvers becomes significantly high, which makes the solver infeasible.

1.1.2 Iterative methods

Iterative methods seek approximate solutions to linear systems via a series of iterations. Some advanced iterative methods are fast, which are more favourable in solving large–size and complicated engineering problems. Straightforward iterative methods are the stationary methods, which have the fixed iteration rule, including the Jacobi, Gauss–Seidel, and successive over–relaxation methods [95, 120]. It is noted that convergence of the stationary methods is conditional, i.e. problem dependent [120].

A more sophisticated family of iterative method are the Krylov-subspace methods, such as the conjugate gradient and the generalised minimum residual methods [95]. For a sparse matrix system, the work cost of each iteration of the conjugate gradient method is $O(n)$. However, the convergence of the conjugate gradient method is dependent on the condition number of the matrix [87]. The closer the condition number is to one, the faster the error of the conjugate gradient decays. Therefore, a (left) preconditioner M can be introduced to transfer the original problem to a preconditioned system

$$M^{-1}Au = M^{-1}b, \tag{1.2}$$

and if $M^{-1}A$ has a smaller condition number than the matrix A, then the conjugate gradient method can converge faster [112]. The construction of the preconditioner M is of great importance in solving complex problems. The key is to find a preconditioner that can bound the condition number of the system. In this case, the conjugate gradient method equipped with such preconditioner will have the $O(n)$ algorithm complexity for solving a SPD and sparse linear system. Two advanced linear solvers have been developed that can be served as satisfactory preconditioners, which are the domain decomposition and the multigrid methods.

1.1.3 Domain decomposition

Domain decomposition methods [103, 34] are a class of methods that solve the linear system by decomposing a large problem into a collection of smaller problems. The decomposition strategy fits well the context of the finite element method via decomposing the domain of the problem into several small subdomains. A feature of the domain decomposition methods is the ease of parallelisation by solving the small problems on all subdomains at one time. Its power to deal with problems with complicated domains is also appealing in engineering simulations.

Domain decomposition ideas have been applied to a wide variety of problems, which result in different types of methods. We introduce three common ones here.

- Schwarz type methods [75, 76] are the simplest domain decomposition algorithms, which are formulated by solving the finite element on each subdomain. The approaches are usually viewed as the generalisation of the block Jacobi and the block Gauss–Seidel methods. The convergence results of the Schwarz type methods are discussed in [49]. It is found that the larger size of the overlap between subdomains is, the more accurate solution can be obtained [34].

- Finite element tearing and interconnecting (FETI) method [40, 39] is based on optimising the solution on interfaces of subdomains, via solving the Lagrange multipliers on the interface, to ensure the continuity of the solution. The FETI method enforces the equality of average of solution across the edges and faces on subdomain interfaces, which is important for parallel scalability for 3D problems. FETI is very suitable in high performance parallel computing for solving the solution on each subdomain at one time without the need to update the solution one by one. A mathematical analysis of the FETI method is provided in [84]. FETI methods can be extended to two–level cases, which results in a FETI-DP method (dual-primal unified FETI method) [39], which is a simplification and a better performing version of FETI.

- The balancing domain decomposition by constraints (BDDC) method [33, 43] defines a preconditioner for the Schur complement of the linear system. BDDC sets coarse grid unknowns by the solution at the corners of subdomains and the solution averages over the edges or the faces of the interface between the subdomains, and minimises the energy function of these coarse grid unknowns. Convergence theories of the BDDC method have been studied in [83, 74]. It is noted that the performance of the BDDC method is the same as the FETI-DP method.

1.1.4 Multigrid

We focus on the multigrid methods [27, 54, 108] in this thesis. Multigrid methods are one of the multilevel methods which employ mesh hierarchies. The method was originated to resolve the issue of smoothing property, which is that the stationary iterative methods remove the high frequency error quickly while the smooth component of error persists [41, 42, 23]. Multigrid applies the smoothers on the coarse grids to get rid of the smooth error. A main advantage of multigrid is the potential $O(n)$ algorithm complexity for solving elliptic type PDEs[108].

There are two common families of multigrid, the geometric multigrid (GMG) and the algebraic multigrid (AMG). In GMG, the mesh hierarchies are provided geometrically, which means the meshes of coarse grids are provided. In AMG, the mesh hierarchies are constructed algebraically, namely, except the geometric finest grid, the coarse grids are constructed from the algebraic information of the fine grid. Two common algebraic multigrid methods include the classical AMG [94] and the smoothed aggregation AMG [110, 109]. Convergence of multigrid has been well studied in [22, 118]. Multigrid methods, both GMG and AMG, have already been largely carried out in engineering simulations [93, 113, 17, 44]. Multigrid is often employed as the preconditioner for the conjugate gradient method [8, 63].

1.2 Presence of low quality meshes

When meshing highly complex engineering geometries, it is inevitable that cell quality will be lower in some regions. Much attention has been paid to improve the mesh in different ways, e.g. advanced meshing techniques [121, 64], and geometric mesh improvements [68, 69, 67]. However, improving defects can be time costly, especially if meshes are generated by one team, and analysed by another. As a result, it is necessary to work with those meshes with most sufficient quality cells and just a small number of low quality cells. Due to the large number of low quality meshes, mesh quality plays an important role in engineering simulations. There are many mesh quality measures invented for different purposes [70, 89] . Commonly used mesh quality measures include the dihedral angle, the radius ratio, and the aspect ratio. It is found in [89] that each mesh quality measure has its own effective region. Some works have already been devoted to the mathematical understanding on the relation between the mesh quality and the approximation property of the finite element method, such as [9, 71, 35]. However, there is still no clear boundary of the mesh quality to determine whether a mesh is of high quality or low quality.

While the computational cost of direct solvers is not affected by the cell quality of grids, the performance of iterative solvers is highly dependent on the mesh quality. A small number

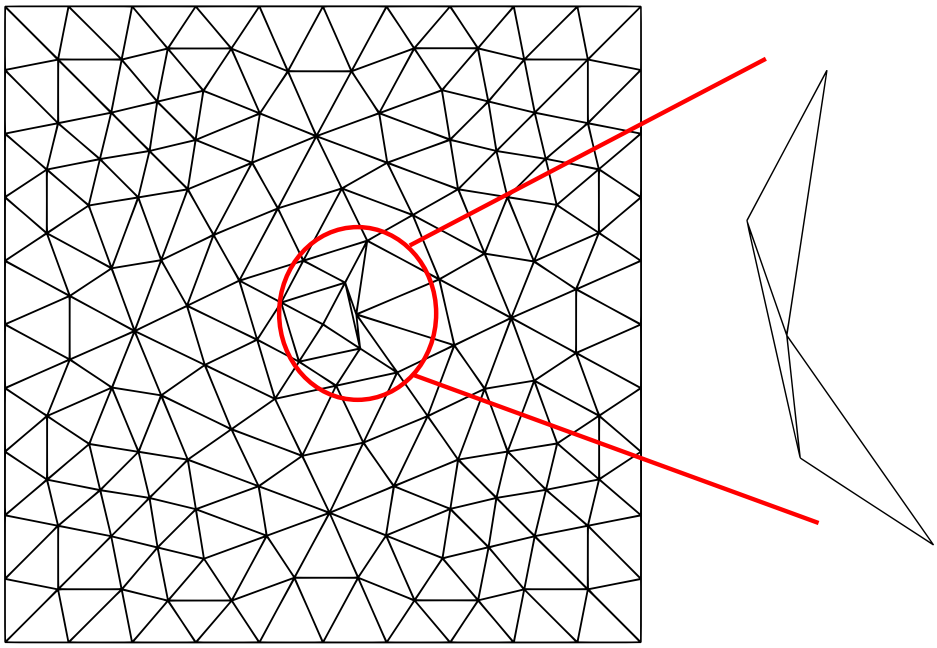


Fig. 1.1 A low quality unit square mesh with one poor quality region at the centre.

of low quality cells can dramatically slow down or even stall the convergence of iterative solvers [102]. In particular, the work [36] employs simple analysis on a structured 2D mesh to show that the standard approximation property of the finite element method still remains valid for those meshes with a small number of local damages, but the condition number of the finite element system becomes arbitrary large, which may lead to the degraded performance of iterative methods. The delay by using the low quality meshes can be substantial and unacceptable in engineering design and analysis process.

1.3 Iterative methods on low quality meshes

We use a unit square mesh with one low quality region at the centre, shown in fig. 1.1, to illustrate the degraded performance of iterative solvers. The low quality triangular cell at the centre has the dihedral angle of one degree. To explore the impact of the mesh quality, we solve the Poisson problem

$$\begin{aligned} -\nabla^2 u &= 2\pi^2 \cos(\pi x)\sin(\pi y) \quad \text{in} \quad \Omega = (0,1)^2 \\ u &= 0 \quad \text{on} \quad \Gamma_D = \{(x,y) \in \partial\Omega : y = 0,1\} \\ \nabla u \cdot n &= 0 \quad \text{on} \quad \Gamma_N = \{(x,y) \in \partial\Omega : x = 0,1\}. \end{aligned} \tag{1.3}$$

on both high and low quality meshes using different iterative methods.

Symmetric Gauss–Seidel, conjugate gradient, geometric multigrid and algebraic multigrid with Gauss–Seidel as smoother are all applied to solve this problem. The convergence results of relative residual at each iteration count are shown in fig. 1.2. It is clear that convergence rate of all above iterative methods is much slower with the low quality mesh compared to the high quality mesh.

1.4 Overview

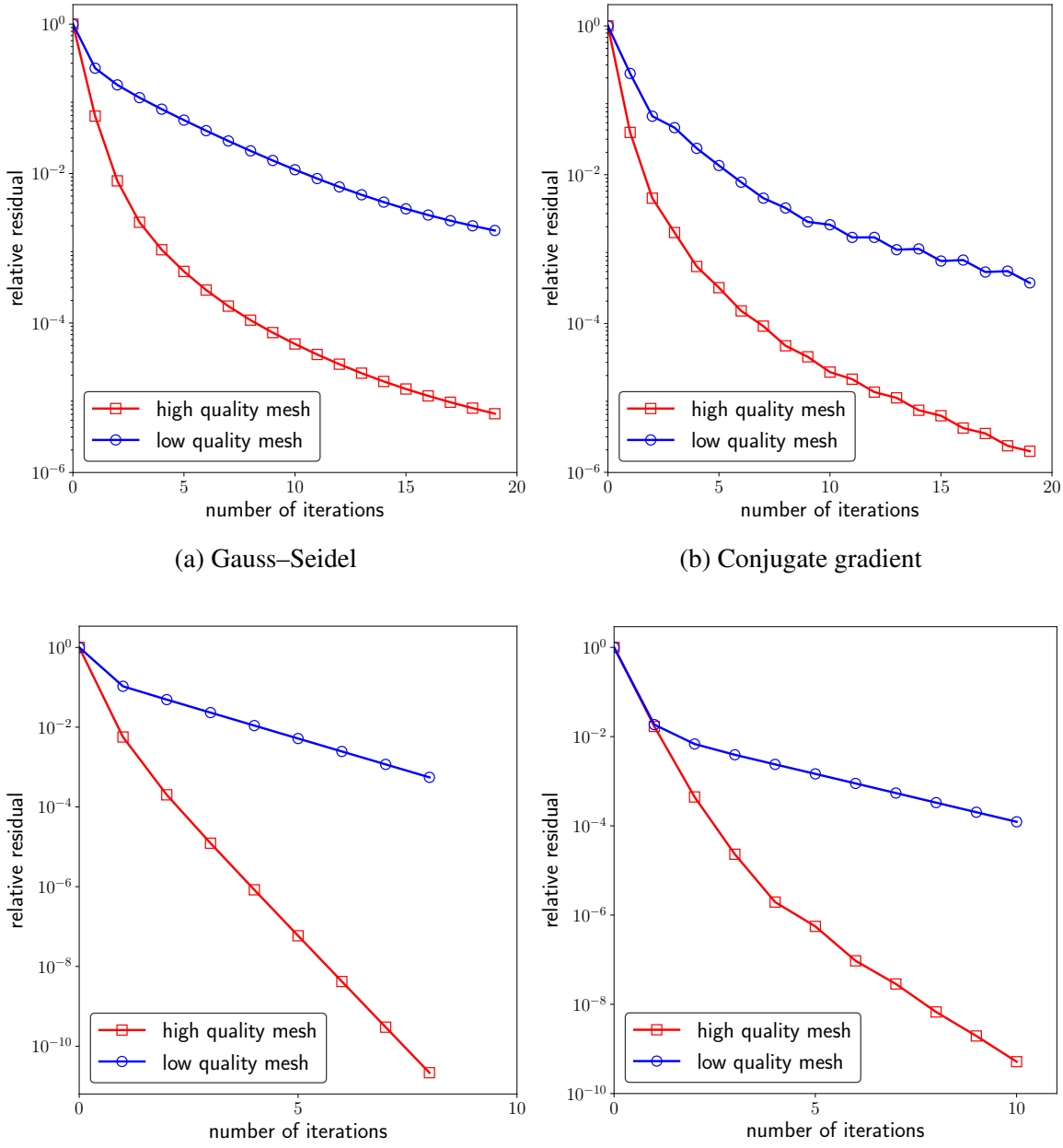
The observation of the poor performance of iterative solvers on low quality meshes leads to our research in this thesis. Our goal is to seek an 'evolution' of multigrid methods that can remedy the poor convergence with low quality meshes to solve elliptic type PDEs including Poisson and elasticity problems. To tackle this problem, we aim to answer the following questions:

- Why does multigrid converge slowly with low quality meshes?
- Is there a way to modify multigrid to deal with meshes in presence of a small number of locally poor quality cells?
- Are we satisfied with the numerical results of using the proposed approaches?
- Can we apply the new method to solve real–world engineering applications?

The thesis address these questions in a logical order. We start with answering the questions for the GMG case, in which we find the reason why multigrid fails with low quality meshes and propose a local correction scheme. Then we shift our discussion to the use of local correction in the AMG case, and both the classical and the smoothed aggregation AMG are considered. A complex turbomachinery problem is also solved to show the effectiveness of the proposed method. The numerical examples in this thesis are produced with freely available and open–source libraries from the FEniCS Project (for implementing finite element methods) [5, 80, 79] and PETSc [11, 10, 12] (for implementing numerical linear solvers). The outline of this thesis is given in the following.

In chapter 2, we give a brief introduction of the finite element method and some useful approximation properties. Some common iterative methods are also introduced in chapter 2. Chapter 3 gives a brief explanation of the multigrid methods with a particular emphasis on using the finite element method. Both GMG and AMG are described with detailed algorithms and formulations given.

From chapter 4 to chapter 6, multigrid methods on low quality meshes are considered. In chapter 4, a local correction scheme is proposed for the geometric multigrid to deal with



(a) Gauss–Seidel

(b) Conjugate gradient

(c) Geometric multigrid

(d) Algebraic multigrid

Fig. 1.2 Convergence of relative residual with iteration counts obtained by different iterative methods acting on the high and low quality unit square meshes.

unstructured meshes containing a small number of locally poor quality cells. Chapter 5 then extends the local correction idea to the case of the classical AMG. We also address the use of the local correction idea for the smoothed aggregation AMG in chapter 6.

Two applications of the multigrid with local correction scheme are presented in chapter 7 and chapter 8. In chapter 7, we apply the new approach to solve the nonlinear problems via the Newton–multigrid method. A thermomechanical simulation of turbomachinery problem is considered in chapter 8, which is successfully solved by our proposed algorithm.

The final chapter, chapter 9, presents conclusions of this work.

Chapter 2

Mathematical background

Numerical methods are essential in engineering simulations. Simulations of engineering problems usually need to approximate solutions to partial differential equations (PDE). A common way to do it is using the finite element method (FEM), which seeks the discrete solution to the weak form of the PDE problem. We focus on solving elliptic type PDEs in this thesis. FEM transforms an elliptic PDE problem to a linear system, thus, a particular problem in engineering is how to solve this linear system accurately and quickly. Iterative solvers have been developed to approximate the solutions to large–size linear systems.

In this chapter, we introduce several mathematical tools needed in this thesis. The finite element method to solve elliptic type PDEs as well as some approximation properties are given in section 2.1. Several iterative solvers are also introduced. Stationary iterative methods to solve the linear systems generated by the finite element method are summarised in section 2.2. The conjugate gradient method is discussed in section 2.3. Preconditioning is introduced in section 2.4, including the preconditioned conjugate gradient algorithm. We also give a brief review of the Chebyshev semi–iterative method in section 2.5. And finally, section 2.6 provides a summary of the Schwarz type domain decomposition methods and some variants including block preconditioners and subspace correction methods.

2.1 Galerkin finite element

The finite element method (FEM) is a widely used technique to approximate the solutions to partial differential equations (PDE) arising in science and engineering. The main feature of the finite element method is its capability to deal with different equations (both linear and nonlinear) and complicated geometries. We give a review here of how FEM approximates the solutions to elliptic type PDEs.

Let the domain of the problem be $\Omega \in \mathbb{R}^{\dim}$, where $\dim = 1,2,3$ is the spatial dimension, and denote the boundary of the domain by $\partial\Omega$. An elliptic PDE problem can be formulated as

$$\begin{aligned} Lu &= f \quad \text{in } \Omega, \\ u &= h \quad \text{on } \Gamma_D, \\ \frac{\partial u}{\partial n} &= g \quad \text{on } \Gamma_N, \end{aligned} \tag{2.1}$$

where L is a uniformly elliptic partial differential operator [26, Equation 5.6.1, 5.6.2], u is the unknown, $f,h,g \in L^2(\Omega)$ are integrable functions on spatial domain, and Γ_D corresponds to the Dirichlet boundaries and Γ_N corresponds to the Neumann boundaries with $\Gamma_D \cup \Gamma_N = \partial\Omega$, and $\Gamma_D \cap \Gamma_N = \emptyset$. We consider the case where L is linear, finite element method for nonlinear PDEs is discussed in chapter 7. To make the illustration simple, the Poisson equation with homogeneous Dirichlet boundary condition

$$\begin{aligned} -\nabla^2 u &= f \quad \text{in } \Omega, \\ u &= 0 \quad \text{on } \partial\Omega, \end{aligned} \tag{2.2}$$

is used to show the discretisation procedure. A test function v is first constructed, which vanishes on the entire boundary, i.e., $v = 0$ on $\partial\Omega$. Multiply the test function to the Poisson equation,

$$-\int_\Omega \nabla^2 u v dx = \int_\Omega f v dx \quad \text{in } \Omega, \tag{2.3}$$

and integrate by parts, we have

$$\int_\Omega \nabla u \cdot \nabla v dx - \int_{\partial\Omega} v \frac{\partial u}{\partial n} ds = \int_\Omega f v dx. \tag{2.4}$$

Inserting the boundary conditions, then

$$\int_\Omega \nabla u \cdot \nabla v dx = \int_\Omega f v dx, \tag{2.5}$$

which is called the weak form of the Poisson equation. In general, the finite element method solves the weak form of the PDE, namely, finds the solution $u \in V$ such that

$$a(u,v) = (f,v) \quad \forall v \in V, \tag{2.6}$$

where v is the test function, and V is a suitable function space. Typically for elliptic PDEs, V is the standard Sobolev space. In the weak form, $a(u,v)$ is a bilinear form in u and v, and (f,v) is a linear form in v. For the case of the Poisson equation, the bilinear and linear forms

become

$$\begin{aligned} a(u,v) &= \int_\Omega \nabla u \cdot \nabla v dx, \\ (f,v) &= \int_\Omega f v dx. \end{aligned} \tag{2.7}$$

The finite element approximates the solution in a finite dimensional space. The so called Ritz-Galerkin FEM discretises the space V, and it seeks the solution in a finite dimensional space $V_h \subset V$, i.e., finds $u_h \in V_h$ such that

$$a(u_h, v_h) = (f, v_h) \quad \forall v_h \in V_h. \tag{2.8}$$

In order to discretise the space, a mesh is created which is a triangulation of the domain Ω. A mesh consists of a collection of non-overlapping cells/elements τ. The union of these cells $\mathscr{T} = \cup\tau$ is the mesh approximating the domain Ω. The cells are usually line segments for 1D, triangles, quadrilaterals for 2D, and tetrahedrals, hexahedrals for 3D domains. In this thesis, simplex cells are used, which refer to line, triangle and tetrahedral elements.

The symbol h in FEM denotes the cell diameter, i.e. the maximal cell size, of all cells. If the sizes of all cells in a mesh are the same, then the mesh is called a uniform mesh; and if the connectivity for cells is also the same, then the mesh is called a structured mesh. Otherwise, the mesh is unstructured. Unstructured meshes are regularly employed in engineering applications.

After the mesh is generated, a set of basis functions $\{\varphi_i\}_{i=1}^n$ is constructed with the linear combination of the basis functions spanning the space V_h,

$$\text{span}\,\{\varphi_i\}_{i=1}^n = V_h. \tag{2.9}$$

The discretised solution u^h can then be written in terms of the basis functions,

$$u_h = \sum_{i=1}^n U_i \varphi_i, \tag{2.10}$$

where $U_i \in \mathbb{R}$ are the unknowns, referred to as degrees-of-freedoms (DOF). The number of DOFs n characteristics the size of the FEM problem. The basis functions used in this thesis are the Lagrange polynomials of low order, in which P_1 and P_2 elements denoting linear and quadratic polynomials respectively. For instance, the plot of a P_1 Lagrange basis for a $2D$ triangular cell is shown in fig. 2.1.

Inserting the discretised field into the weak form leads to a linear system of n algebraic equations with n unknowns

$$Au = b, \tag{2.11}$$

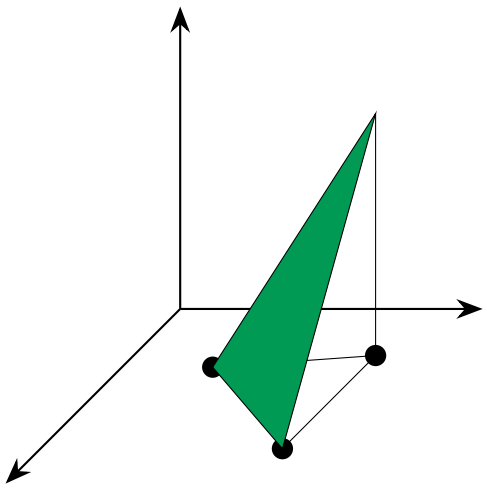


Fig. 2.1 A P_1 Lagrange basis function for a $2D$ triangular cell.

where $A \in \mathbb{R}^{n\times n}$ is known as the 'stiffness matrix', $b \in \mathbb{R}^n$ is called the 'right-hand side', and $u \in \mathbb{R}^n$ is the unknown field containing all DOFs to solve. In this thesis, we focus on the multigrid method to solve the algebraic system eq. (2.11). Multigrid is particularly suitable for solving large size elliptic type PDEs, thus only elliptic type problems are considered here. However, by no means can multigrid only solve the elliptic problems, it has been applied to many different equations. A detailed description can be found in chapter 3.

There is extensive mathematical analysis on the finite element method, including existence and uniqueness, accuracy, and regularity of results. We refer the readers to some classical texts [26, 30, 19]. We state the relevant convergence theorem here which is the *a priori* error estimate. The *a priori* error estimate is one of the most important convergence results for FEM solutions, which relates the error to the mesh size and polynomial order of the basis functions. Let us define some useful function spaces and their norms first.

Definition 2.1. *The Lebesgue space $L^2(\Omega)$ contains functions that are second order integrable,*

$$L^2(\Omega) = \{u : \|u\|_{L^2(\Omega)} < \infty\} \tag{2.12}$$

where the associated norm is

$$\|u\|_{L^2(\Omega)} = \left(\int_\Omega |u(x)|^2 dx\right)^{1/2}. \tag{2.13}$$

The Sobolev space $H^m(\Omega)$ is a subspace of $L^2(\Omega)$ defined as

$$H^m(\Omega) = \{u : u \in L^2(\Omega) | \, \partial^\alpha u \in L^2(\Omega), |\alpha| \leq m\} \tag{2.14}$$

the associated norm is

$$\|u\|_{H^m(\Omega)} = \left(\sum_{\|\alpha\| \leq m} \|\partial^\alpha u\|^2_{L^2(\Omega)} \right)^{1/2} \tag{2.15}$$

and the semi-norm is

$$|u|_{H^m(\Omega)} = \left(\sum_{\|\alpha\| = m} \|\partial^\alpha u\|^2_{L^2(\Omega)} \right)^{1/2}. \tag{2.16}$$

The *a priori* error estimate is given in the next theorem.

Theorem 2.1. *Let u_h satisfy* (2.8) *for a second–order uniformly elliptic PDE with $u_h \in V := H^m(\Omega)$, and suppose u is the exact solution to* (2.6)*, then we have*

$$\|u - u_h\|_{H^m(\Omega)} \leq Ch^{p+1-m} |u|_{H^{p+1}(\Omega)}, \tag{2.17}$$

where $p \geq 1$ is the polynomial order of the basis functions, and h is the maximum cell size in the mesh.

The detailed proof of the *a priori* error estimate can be found in [26, Theorem 4.7.3]. Moreover, the two main results we are interested are related to H^1 and L^2 norm,

$$\begin{aligned} \|u - u_h\|_{H^1(\Omega)} &\leq Ch^p |u|_{H^{p+1}(\Omega)}, \\ \|u - u_h\|_{L^2(\Omega)} &\leq Ch^{p+1} |u|_{H^{p+1}(\Omega)}. \end{aligned} \tag{2.18}$$

2.2 Stationary iterative methods

In this section, we consider some classical methods to solve the linear system generated by finite element method

$$Au = b \tag{2.19}$$

where $A \in \mathbb{R}^{n \times n}$. Specifically, when using finite element discretisation to solve uniformly elliptic PDEs, then the discrete system A is symmetric positive-definite [26, Section 5.6], which is defined in the following.

Definition 2.2. *A matrix A is symmetric positive-definite(SPD) if it is symmetric $A = A^T$, and*

$$v^T A v > 0, \quad \forall v \in \mathbb{R}^n \setminus 0. \tag{2.20}$$

Therefore, in this chapter, numerical solvers are introduced to solve the SPD matrices.

A direct solver, such as *LU* factorisation, can be applied to solve the above problem. The major drawback of the direct solver is its algorithmic complexity and related computational cost. A sparse direct solver usually has algorithmic cost complexity that is a power of n, with the exponent depending on the spatial dimension. For structured meshes, direct solver requires at least $O(n^{3/2})$ time and $O(n\log n)$ storage for two dimensional problems, and $O(n^2)$ time and $O(n^{4/3})$ storage for three dimensional problems [47]. Even if applying high–level parallel computing, the high computational cost with problem size can never be conquered. In many cases, especially for engineering applications, to reduce the error, a smaller size of element is needed, which would cause the direct solver not feasible.

Iterative methods create a successive sequence of approximate solutions $u_1, u_2, \cdots, u_k$ that, hopefully, converge to the exact solution. These methods are potentially efficient in terms of computation for large sparse linear system, in which the matrix A contains a high percentage of zero entries. We first define the residual and error for an approximated solution.

Definition 2.3. *If u_k is the k-th approximation to the linear system eq.* (2.19)*, and u^* is the exact solution to the linear system, then the residual r_k and error e_k at k-th iteration are given by*

$$\begin{aligned} r_k &= b - Au_k, \\ e_k &= u^* - u_k. \end{aligned} \tag{2.21}$$

Stationary iterative methods involve a linear update rule that is fixed during iterations. These methods are normally generated by operator splitting strategy. Suppose we split the matrix A such that

$$A = M + K, \tag{2.22}$$

where M is nonsingular. Then, eq. (2.19) becomes

$$\begin{aligned} Mu &= -Ku + b, \\ Mu - Au &= -Ku + b - Au, \\ u &= u + M^{-1}(b - Au). \end{aligned} \tag{2.23}$$

Given an initial approximation, the fixed equation can be iteratively computed

$$u_{k+1} = u_k + M^{-1} r_k, \tag{2.24}$$

which is called the residual representation of the iterative method. Rearranging the residual representation formula, we get

$$u_{k+1} = Tu_k + c, \tag{2.25}$$

where $T = -M^{-1}K$ and $c = M^{-1}b$, which is the iterative representation of the method.

Iterative methods	M	K	T	c
Richardson	ωI	$A-\omega I$	$I-\omega^{-1}A$	$\omega^{-1}b$
Jacobi	D_A	L_A+U_A	$-(L_A+U_A)D_A^{-1}$	$D_A^{-1}b$
Gauss-Seidel	D_A+L_A	U_A	$-(D_A+L_A)^{-1}U_A$	$(D_A+L_A)^{-1}b$

Table 2.1 Splitting matrices for different stationary iterative methods.

The idea behind these methods is to have the matrix M easier to invert than A. Three classical stationary iterative methods are the Richardson, Jacobi and Gauss–Seidel methods, which are based on the partitioning of matrix A. Let D_A, L_A, and U_A be the diagonal, strictly lower triangular, and strictly upper triangular matrices of A respectively, then

$$A = D_A + L_A + U_A \tag{2.26}$$

which is illustrated in fig. 2.2. The definitions of these methods are shown in the table 2.1.

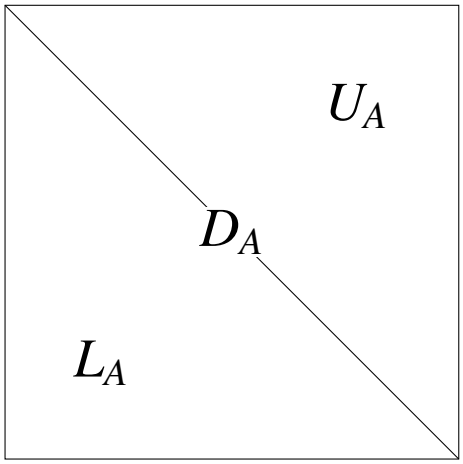


Fig. 2.2 Partitioning of a nonsingular matrix A.

For the classical Richardson method, it is equipped with $M = I$. However, it is found that if inserting a constant ω with $M = \omega I$, the convergence can be faster. The optimal value is $\omega = 2/(\lambda_{\min}(A) + \lambda_{\max}(A))$ for solving a SPD matrix A, where $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ are the smallest and the largest eigenvalues of A respectively [95, Example 4.1].

The convergence of stationary iterative methods has been well studied, and many classical texts provide detailed explanation and analysis on these methods, e.g., [111, Chap. 3; 56, Chapter 2] The stationary methods usually converge conditionally, which means the convergence depends on properties of the original matrix. Let us define some concepts in matrix algebra.

Definition 2.4. *The spectral radius of the matrix is the largest absolute value of the eigenvalues,*

$$\rho(A) = \max\{|\lambda_1|, |\lambda_2|, \cdots, |\lambda_n|\}. \tag{2.27}$$

The condition number (in 2-norm) is defined as

$$\kappa_2(A) = \frac{|\lambda_{\max}(A)|}{|\lambda_{\min}(A)|}. \tag{2.28}$$

Via simple analysis, eigenvalues of a SPD matrix are nonnegative real numbers [120, Section 1.3]. Thus, the condition number for a SPD matrix is the ratio of the largest eigenvalue and the smallest eigenvalue, then the more clustered eigenvalues of a SPD system are, the smaller the condition number is. The main convergence theory of stationary iterative methods depends on the spectral radius, which is shown in the following theorem.

Theorem 2.2. *For any initial guess u_0, the stationary iterative method in* (2.25) *to solve a SPD matrix converges to a unique solution if $\rho(T) < 1$, where $\rho(T)$ is the spectral radius of the iterative matrix T.*

We now consider another two advanced methods in more detail. The first is the weighted or damped Jacobi method. The weighted Jacobi is defined by

$$\begin{aligned} M &= \omega D_A \\ T &= I - \omega {D_A}^{-1} A, \end{aligned} \tag{2.29}$$

where ω is also an accelerating parameter, usually set as

$$\omega = \frac{4}{3\lambda_{\max}} D_A^{-1} A, \tag{2.30}$$

where $\lambda_{\max}$ is taken as $\rho(D_A^{-1}A)$. Weighted Jacobi will be used in the smoothed aggregation AMG setting later in section 3.6.2. Another method is the symmetric Gauss–Seidel. It is noted that the Gauss–Seidel operator is not symmetric even if A is symmetric. The symmetric Gauss–Seidel is made of a forward–backward procedure of the original Gauss–Seidel, which results in the following iterative operator:

$$M = (D_A + L_A) D_A^{-1} (D_A + U_A). \tag{2.31}$$

2.3 Conjugate gradient method

The conjugate gradient method is one of the Krylov subspace methods. The technique approximates the solution in a finite dimensional subspace called the Krylov subspace, defined as

$$\mathscr{K}_k(A,b) = \text{span}\{b, Ab, \cdots, A^{k-1}b\}, \tag{2.32}$$

or

$$\mathscr{K}_k(A,r_0) = \text{span}\{r_0, Ar_0, \cdots, A^{k-1}r_0\}. \tag{2.33}$$

The approximate solution u_k is constructed from the affine subspace $u_{k-1} + \mathscr{K}_k(A,b)$. Then the problem becomes how to choose the combination in $\mathscr{K}_k$, with which the 'best' solution can be obtained. The conjugate gradient method chooses to require the orthogonality of residuals, i.e.,

$$r_j^T r_k = 0, \quad \forall j < k. \tag{2.34}$$

It is noticed that $u_j - u_{j-1} \in \mathscr{K}_j$, which leads to

$$\left(u_j - u_{j-1}\right)^T (r_k - r_{k-1}) = 0, \quad \forall j < k. \tag{2.35}$$

Therefore, conjugate gradient computes the difference $\Delta u = u_k - u_{k-1}$ by

$$\left(u_j - u_{j-1}\right)^T A\,(u_k - u_{k-1}) = 0, \quad \forall j < k. \tag{2.36}$$

Algorithm 2.1 illustrates the conjugate gradient method for solving the linear system $Au = b$.

Algorithm 2.1 Conjugate gradient $u_k = cg(A, b, u_{k-1}, s_{k-1})$

1: **for** $k = 1, 2, \cdots$ **do**
2: Start with $s_0 = r_0$
3: Set $\alpha_k = \frac{r_{k-1}^T r_{k-1}}{s_{k-1}^T A s_{k-1}}$.
4: Update the solution by $u_k = u_{k-1} + \alpha_k s_{k-1}$.
5: Find the residual $r_k = r_{k-1} - \alpha_k A s_{k-1}$.
6: Define $\beta_k = \frac{r_k^T r_k}{r_{k-1}^T r_{k-1}}$.
7: Update the searching direction by $s_k = r_k + \beta_k s_{k-1}$.

A key feature of the conjugate gradient method is that it minimises the variational functional of the linear system

$$F(u) = \frac{1}{2} u^T A u - u^T b, \tag{2.37}$$

in the Krylov subspace. The exact solution is the true minimiser to the functional, while if u_k is the k-th iteration output of the conjugate gradient method, then

$$u_k = \arg\min_{v \in \mathscr{K}_k} F(v). \tag{2.38}$$

The convergence of conjugate gradient method is related to the condition number of the system. A detailed analysis can be found in [87]. The best-known result is the following theorem.

Theorem 2.3. *Let A be a symmetric positive-definite matrix, and u_k be the k-th iteration solution of conjugate gradient method with corresponding error e_k, then*

$$\|e_k\|_A \leq 2\sqrt{\kappa_2(A)} \left(\frac{\sqrt{\kappa_2(A)} - 1}{\sqrt{\kappa_2(A)} + 1} \right)^k \|e_0\|_A \tag{2.39}$$

From the above convergence theorem, the closer the condition number is to one, the faster the error of the conjugate gradient decays. Another feature of the conjugate gradient method is its algorithm complexity. Suppose we are solving a sparse matrix system, which technically means that the total number of nonzero entries of the matrix is roughly less than the matrix size, so that the sparse matrix–vector multiplication is of $O(n)$. In the algorithm of conjugate gradient, the main work for each iteration is two matrix-vector multiplications. Thus, the work cost of each iteration for the conjugate gradient is $O(n)$ for solving a sparse matrix if the condition number of the system is uniformly bounded, and moreover, the method would converge in a number of iterations that is independent of the problem size.

There are many other types of Krylov subspace method, the commonly used ones are

- Arnodi and generalized minimal residual method (GMRES) [7, 96]: residual r_k has the minimum norm for u_k in $\mathscr{K}_k(A)$.
- Biconjugate gradient [91]: residual r_k is orthogonal to $\mathscr{K}_k(A^T)$.
- Lanczos iteration [72]: approximate eigenvalues from $\mathscr{K}_k(A)$.

2.4 Preconditioning

The idea of preconditioning comes from two aspects. First, the stationary iterative methods are not fast enough, which may need a large number of iterations to converge. A preconditioner can help accelerate these methods. On the other hand, we have already seen that the convergence of the conjugate gradient method depends on the condition number of the linear

system, and if a better conditioned system can be obtained, then the system is much easier to be solved by the conjugate gradient. A preconditioner is defined as a nonsingular operator M which has the same size of A. Instead of solving the original linear system, it can be more efficient to solve the system

$$M^{-1}Au = M^{-1}b, \tag{2.40}$$

if the condition number of this system is smaller, i.e., $\kappa_2(M^{-1}A) < \kappa_2(A)$.

The Jacobi and Gauss–Seidel methods can then be viewed as the preconditioned Richardson methods with

$$\begin{aligned} u_{k+1} &= u_k + M^{-1}r_k \\ M &= D_A, \qquad \text{Jacobi} \\ M &= (D_A + L_A), \qquad \text{Gauss–Seidel.} \end{aligned} \tag{2.41}$$

An important solver is the preconditioned conjugate gradient method. A good preconditioner can significantly reduce the condition number of the original system, which results in a more suitable system for the use of the conjugate gradient method. Suppose there is some preconditioner M, which is nonsingular, such that $M^{-1} \approx A^{-1}$, i.e., $M^{-1}b$ also approximates the solution to the system, then the condition number of $M^{-1}A$ gets much closer to one. The residual of the preconditioned system becomes $z_k = M^{-1}r_k$, where r_k is the original residual. For the preconditioned conjugate gradient method, the residuals of the preconditioned system eq. (2.40) are kept orthogonal to each other. Algorithm 2.2 explains the details of the preconditioned conjugate gradient method. In terms of implementation, M^{-1} is not an

Algorithm 2.2 Preconditioned conjugate gradient $u_k = pcg(A, b, M^{-1}, u_{k-1}, s_{k-1})$

1: **for** $k = 1, 2, \cdots$ **do**
2: Compute $r_0 = b - Au_0$, $z_0 = M^{-1}r_0$, and $p_0 := z_0$.
3: Set $\alpha_k = \frac{r_{k-1}^T z_{k-1}}{p_{k-1}^T A p_{k-1}}$.
4: Update the solution by $u_k = u_{k-1} + \alpha_k p_{k-1}$.
5: Find the residual $r_k = r_{k-1} - \alpha_k A p_{k-1}$.
6: Solve the system $z_k = M^{-1}r_k$.
7: Define $\beta_k = \frac{r_k^T z_k}{r_{k-1}^T z_{k-1}}$.
8: Update the searching direction by $p_k = z_k + \beta_k p_{k-1}$.

explicit matrix form but an algorithm to invert the original system. For example, the Jacobi preconditioned conjugate gradient method stands for using the Jacobi method to find M^{-1}.

2.5 Chebyshev semi-iterative method

The Chebyshev method is a polynomial iterative method, which can be viewed as a semi-iterative scheme. The so called polynomial method is an acceleration strategy for stationary iterative methods. Some classical texts containing the details of semi-iterative methods and their convergence are [56, Chapter 3–4].

Suppose w_k has already been obtained by some stationary iterative method in eq. (2.25) via

$$w_{j+1} = Tw_j + c \quad j = 0, 1, \cdots, k-1, \tag{2.42}$$

where the iterative operator T is defined in table 2.1. To speed up the convergence, an improving approximation is obtained with

$$u_k = \sum_{j=0}^{k} \alpha_{k,j} w_j, \tag{2.43}$$

where $\alpha_{k,j}$ are real numbers subject to the constraint

$$\sum_{j=0}^{k} \alpha_{k,j} = 1, \tag{2.44}$$

to ensure the final convergence [56]. In terms of residual representation, we write eq. (2.43) as

$$\begin{aligned} w_{j+1} &= w_j + Mr_j, \quad j = 0, 1, \cdots, k-1 \\ u_k &= u_{k-1} + P_M(b - Aw_k), \end{aligned} \tag{2.45}$$

where P_M is some polynomial of the original preconditioner M,

$$P_M = \sum_{j=0}^{k} \alpha_{k,j} M^j. \tag{2.46}$$

It is clear that the preconditioner operator becomes a polynomial applied on some stationary iterative operator M, which is why we call it a polynomial method.

The error representation of the polynomial method can be expressed as

$$e_k = q_k(T) e_0 \tag{2.47}$$

where

$$q_k(T) = \alpha_{k,0} I + \alpha_{k,1} T + \cdots + \alpha_{k,k} T^k. \tag{2.48}$$

The Chebyshev method minimises the error by taking a special $q_k(T)$ which is related to the Chebyshev polynomial.

Definition 2.5. *Chebyshev polynomial $Q_n(x)$ [56] can be generated by the following recurrence relation*

$$\begin{aligned} Q_0(x) &= 1, \\ Q_1(x) &= x, \\ Q_{n+1}(x) &= 2xQ_n(x) - Q_{n-1}(x), \quad n \geq 1, \end{aligned} \tag{2.49}$$

and its algebraic expression is given by

$$Q_n(x) = \begin{cases} \cosh(n\cosh^{-1}x), & x > 1 \\ \cos(n\cos^{-1}x), & -1 \leq x \leq 1 \\ (-1)^n\cosh(n\cosh^{-1}(-x)), & x < -1. \end{cases} \tag{2.50}$$

The polynomial accelerator $q_k(T)$ is based on the property of the Chebyshev polynomial and defined via

$$q_k(T) = Q_k\left(\frac{2T - \lambda_{\max}(T) - \lambda_{\min}(T)}{\lambda_{\max}(T) - \lambda_{\min}(T)}\right) \Big/ Q_k\left(\frac{2 - \lambda_{\max}(T) - \lambda_{\min}(T)}{\lambda_{\max}(T) - \lambda_{\min}(T)}\right), \tag{2.51}$$

where Q_k is the Chebyshev polynomial, and $\lambda_{\max}$ and $\lambda_{\min}$ are the largest and smallest eigenvalues of the standard iterative system T. With this accelerator polynomial, it is found that the error produced by the Chebyshev method is $O(\rho(P_k(T))$ [56]. The Chebyshev method is well-known for its strong potential of parallel implementation [51, 1]. In many real engineering problems, the Jacobi preconditioned Chebyshev method is largely used for its simplicity and efficiency. In this thesis, the Jacobi preconditioned Chebyshev method is employed as a smoother for multigrid. It is emphasized here that the extreme eigenvalues used in constructing the Chebyshev polynomial parameters in eq. (2.51) can influence the convergence rate of the Chebyshev method. Some rigorous bounds have been established in [56] for estimating the convergence error caused by the inaccuracy in approximating the largest and smallest eigenvalues. Moreover, in [1], it suggests that the use of the Chebyshev method only requires the largest eigenvalue, and the smallest eigenvalue can be approximated as a fraction of the largest eigenvalue.

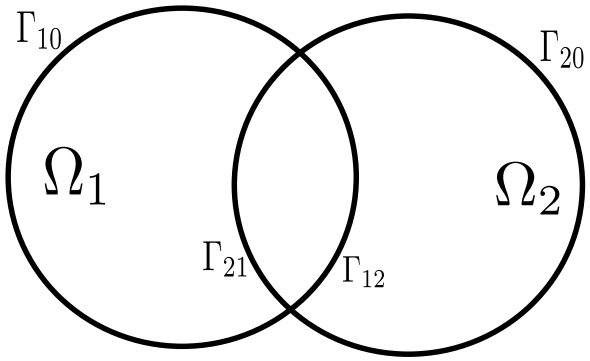


Fig. 2.3 Partition of the domain into two subdomains.

2.6 Simple domain decomposition methods

Domain decomposition methods [103, 34] are classes of iterative methods for solving large linear systems generated from elliptic PDEs in parallel machines. Instead of solving the one linear system, several smaller systems are solved with the smaller systems generated by decomposing the domain into several subdomains. The simplest domain decomposition methods are the Schwarz type methods. And, we also introduce the block preconditioners and the subspace correction methods here, which are algebraically equivalent to the Schwarz type domain decomposition methods, while they differ from the way how to make the decomposition.

2.6.1 Schwarz domain decomposition methods

Schwarz methods [103, 34] are the most straightforward domain decomposition methods, which are one of the one level algorithms. These methods were developed before the presence of modern functional analysis on PDE like the Sobolev space. They are usually not the most efficient methods, typically may not converge in a small number of iterations, but aid in understanding more complicated multilevel methods.

Let the domain Ω decompose into D subdomains (possibly overlapping) with

$$\Omega = \bigcup_{d=1}^{D} \Omega_d, \tag{2.52}$$

as illustrated in fig. 2.3. We use Γ_{d0} to denote the boundary of the subdomain Ω_d which is also on the boundary of the entire domain, i.e., $\Gamma_{d0} = \partial\Omega_d \cap \partial\Omega$. The boundary of the subdomain Ω_d that is included in subdomain Ω_j is denoted by Γ_{dj} with $\Gamma_{dj} = \partial\Omega_d \cap \overline{\Omega_j}$.

Let our problem be a linear PDE

$$\begin{aligned} Lu &= f \quad \text{in } \Omega, \\ u &= g \quad \text{on } \partial\Omega, \end{aligned} \tag{2.53}$$

where L is a linear elliptic operator. The most straightforward approach is the additive Schwarz method, for $d = 1, 2, \cdots, D$

$$\begin{aligned} Lu_{k+1}^d &= f \quad \text{in } \Omega_d \\ u_{k+1}^d &= g|_{\Gamma_{d0}} \quad \text{on } \Gamma_{d0} \\ u_{k+1}^d &= u_k^d|_{\Gamma_{dj}} \quad \text{on } \Gamma_{dj}, \end{aligned} \tag{2.54}$$

in which we solve the PDE on each subdomain alternatively. Let the size of each subsystem be n_d. Let the operator I_d be the interpolation mapping from the subdomain Ω_d to the whole domain Ω, which is a rectangular Boolean matrix with size $n \times n_d$. The restriction operator R_d mapping from the whole domain to the subdomain is then given by the transpose of the interpolation operator, i.e. I_d^T. In this scheme, the boundary conditions imposed on each subdomain are essential. However, if we rearrange the formula eq. (2.54), and consider the (continuous) residual function $r = f - Lu$, then the additive Schwarz algorithm is equivalent to solve the error on each subdomain,

$$\begin{aligned} r_k &= f - Lu_k \\ Le_{k+1}^d &= r_k|_{\Omega_d} = I_d^T r_k \quad \text{in } \Omega_d \\ e_{k+1}^d &= 0 \quad \text{on } \partial\Omega_d \\ u_{k+1} &= u_k + \sum_{d=1}^{D} I_d e_{k+1}^d, \end{aligned} \tag{2.55}$$

which is the same as imposing homogeneous Dirichlet boundary condition on the boundary of each subdomain. In terms of the discrete algebraic formulation, the subsystem defined on the subdomain Ω_d is given by

$$A_d = I_d^T A I_d. \tag{2.56}$$

Inserting the subsystem eq. (2.56) to the algorithm eq. (2.54), denote update on the d-th subdomain as $u_{k+d/D}$, for $d = 1, 2, \cdots, D$, then each iteration on one subdomain can be formulated as

$$u_{k+1/d} = u_k + I_d (I_d^T A I_d)^{-1} I_d^T r_k. \tag{2.57}$$

Thus, the preconditioner form of the additive Schwarz method can be derived by summing over the procedures on each subdomain,

$$\begin{aligned} u_{k+1} &= u_k + \sum_{d=1}^{D} M_d r_k \\ M_d &= I_d (I_d^T A I_d)^{-1} I_d^T. \end{aligned} \tag{2.58}$$

On the other hand, if we update the residual each time after the error on one subdomain is calculated, then a multiplicative Schwarz method is obtained with

$$\begin{aligned} L e_{k+1}^d &= r_k|_{\Omega_d} = I_d^T (f - L u_k) \\ e_{k+1}^d &= 0 \quad \text{on } \partial\Omega_d \\ u_{k+1} &= u_k + I_d^T e_{k+1}^d, \end{aligned} \tag{2.59}$$

for $d = 1, 2, \cdots, D$. Each substep can be written algebraically via

$$\begin{aligned} u_{k+1/D} &= u_k + I_1^T (I_1^T A I_1)^{-1} I_1^T (b - A u_k) \\ u_{k+2/D} &= u_{k+1/D} + I_2^T (I_2^T A I_2)^{-1} I_2^T (b - A u_{k+1/D}) \\ &\cdots \\ u_{k+1} &= u_{k+(D-1)/D} + I_D^T (I_D^T A I_D)^{-1} I_D^T (b - A u_{k+(D-1)/D}). \end{aligned} \tag{2.60}$$

The preconditioner from of multiplicative Schwarz method can then be expressed as

$$\begin{aligned} u_{k+1} &= u_k + M r_k \\ M &= \left[I - (I - M_1 A)(I - M_2 A) \cdots (I - M_D A) \right] A^{-1} \end{aligned} \tag{2.61}$$

It is noted that the additive Schwarz is symmetric, but even when the system A is symmetric, the multiplicative Schwarz operator is not symmetric. Similar to the symmetric Gauss–Seidel method, we can symmetrise the method by including a backward process. Let us use a two–subdomain case to illustrate. The preconditioner form of the multiplicative Schwarz method on a two-subdomain case is $M_1 + M_2 - M_2 A M_1$, which is not symmetric. If a third step is included via

$$\begin{aligned} u_{k+1/3} &= u_k + M_1 (b - A u_k) \\ u_{k+2/3} &= u_{k+1/3} + M_2 (b - A u_{k+1/3}) \\ u_{k+1} &= u_{k+2/3} + M_1 (b - A u_{k+2/3}), \end{aligned} \tag{2.62}$$

then it results in a symmetric preconditioner operator

$$M = M_1 + (I - M_1 A)M_2(I - M_1 A). \tag{2.63}$$

Convergence of the Schwarz type methods is dependent on some approximation properties of the PDE problem. However, a general thought is that the Schwarz type methods are still standard linear iterative methods so that we can apply the convergence condition of stationary iterative methods in theorem 2.2. In order to ensure convergence, we can still consider the requirement on the spectral radius. A detailed proof of convergence can be found in [103, Chapter 5]. It is noted in [34, Secion 1.5] that the convergence of the Schwarz type methods depends on the size of overlapping area between subdomains. If the size of the overlapping area becomes larger, a faster convergence can be obtained.

2.6.2 Block preconditioners

The block preconditioner methods including block Jacobi and block Gauss-Seidel are variants of regular (point) Jacobi and Gauss-Seidel methods. They apply stationary iterative methods on block systems which are typically submatrices of the target matrix. These submatrices are generated from splitting the matrix in block form. Let us partition the set of indices $N = \{1, 2, \cdots, n\}$ into D sets N_d with

$$\{1, 2, \cdots, n\} = \bigcup_{d=1}^{D} N_d. \tag{2.64}$$

Let the interpolation operator be $I_d : N_d \to N$ with

$$I_d[i, j] = \begin{cases} 1 & i = n_j,\ n_j \in N_d \\ 0 & \text{Otherwise} \end{cases}. \tag{2.65}$$

The corresponding solution on each block is $U_d = u|_{N_d} = I_d^T u$, and the right hand side is $b_d = b|_{N_d} = I_d^T b$. We decompose the linear system $Au = b$ into block matrices

$$\begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1D} \\ A_{21} & A_{22} & \cdots & A_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ A_{D1} & A_{D2} & \cdots & A_{DD} \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ \vdots \\ U_D \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_D \end{bmatrix}, \tag{2.66}$$

where A_{ij} refers to one of the small blocks of the linear system A, formulated by

$$A_{ij} = I_i^T A I_j. \tag{2.67}$$

Similar to the point Jacobi method, the block Jacobi solves the diagonal parts of the block system via

$$\begin{bmatrix} A_{11} & 0 & \cdots & 0 \\ 0 & A_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_{DD} \end{bmatrix} \begin{bmatrix} U_{k+1}^1 \\ U_{k+1}^2 \\ \vdots \\ U_{k+1}^D \end{bmatrix} = \begin{bmatrix} A_{11} & 0 & \cdots & 0 \\ 0 & A_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_{DD} \end{bmatrix} \begin{bmatrix} U_k^1 \\ U_k^2 \\ \vdots \\ U_k^D \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_D \end{bmatrix} - A \begin{bmatrix} U_k^1 \\ U_k^2 \\ \vdots \\ U_k^D \end{bmatrix}, \tag{2.68}$$

or equivalently,

$$\begin{bmatrix} U_{k+1}^1 \\ U_{k+1}^2 \\ \vdots \\ U_{k+1}^D \end{bmatrix} = \begin{bmatrix} U_k^1 \\ U_k^2 \\ \vdots \\ U_k^D \end{bmatrix} + \begin{bmatrix} A_{11}^{-1} & 0 & \cdots & 0 \\ 0 & A_{22}^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_{DD}^{-1} \end{bmatrix} \begin{bmatrix} R_k^1 \\ R_k^2 \\ \vdots \\ R_k^D \end{bmatrix}, \tag{2.69}$$

where R_k^d is the block residual with $R_k^d = r_k|_{N_d}$. Thus, the block Jacobi can be easily written as

$$u_{k+1} = u_k + \left(\sum_{d=1}^{D} I_d (I_d^T A I_d)^{-1} I_d^T \right) r_k. \tag{2.70}$$

At this point, it is clear that the block Jacobi is algebraically equivalent to the additive Schwarz method, while N_d becomes the set of indices/DOFs in the subdomain Ω_d.

The block Gauss-Seidel method updates the solution at each time inverting the submatrix, namely, for $d = 1, 2 \cdots, D$,

$$A_{dd} U_{k+1}^d = \begin{bmatrix} 0 & A_{12} & \cdots & A_{1D} \\ A_{21} & 0 & \cdots & A_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ A_{D1} & A_{D2} & \cdots & 0 \end{bmatrix} \begin{bmatrix} U_{k+1}^1 \\ U_{k+1}^2 \\ \vdots \\ U_{k+1}^{d-1} \\ U_k^d \\ \vdots \\ U_k^D \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_D \end{bmatrix}. \tag{2.71}$$

Rearranging the above formula, it is easy to see that the block Gauss–Seidel is algebraically equivalent to the multiplicative Schwarz method. The key to the block preconditioners is that the submatrix to be solved is obtained by cutting the original matrix into small pieces with respect to some sets of indices/DOFs.

2.6.3 Subspace correction

In [116, 117], Xu proposes a subspace correction method, and uses this idea to generalise the multigrid method. The subspace correction method is based upon decomposing the whole space (finite element function space) into a sum of smaller subspaces. Let V be the finite element space of the problem. Suppose that there are D subspaces $V_d \in V$ such that

$$V = \sum_{d=1}^{D} V_d. \tag{2.72}$$

The idea of this method is to solve the corresponding smaller system on each subspace. The subsystem A_d is interpreted as the restriction of A on V_d by the orthogonal projections. Let I_d be the natural inclusion with $I_d : V_d \to V$ such that

$$I_d v = v \quad \forall v \in V_d. \tag{2.73}$$

The subsystem is then formulated as

$$A_d = I_d^T A I_d. \tag{2.74}$$

If the solution to the linear system is u, then $u_d = I_d^T u$ is the restriction of solution u to the subspace V_d, which satisfies

$$A_d u_d = b_d := I_d^T b. \tag{2.75}$$

Moreover, it is found that the subsolution can be the minimiser of the variational form in the subspace with

$$u_d = \underset{v_d \in V_d \subset V}{\arg\min} F(v_d), \tag{2.76}$$

where F is the variational form defined in eq. (2.37). The system to be solved on each subspace is the residual equation,

$$A_d e_d = r_d, \tag{2.77}$$

which makes the method a correction method. If we solve the residual equation on each subspace, and update the solution in order, we get

$$u_{k+1} = u_k + \sum_{d=1}^{D} I_d e_d, \tag{2.78}$$

which leads to the parallel subspace correction method. The residual representation of the parallel subspace correction is then given by

$$\begin{aligned} u_{k+1} &= u_k + \sum_{d=1}^{D} M_d r_k \\ M_d &= I_d (I_d^T A I_d)^{-1} I_d^T. \end{aligned} \tag{2.79}$$

If we solve the residual equation on one subspace at a time by using the most recently updated approximation of u, then the successive subspace correction method is obtained. Suppose in each iteration from u_k to u_{k+1}, there are D steps from $u_{k+(d-1)/D}$ to $u_{k+d/D}$, for $d = 1, 2, \cdots, D$, the iterative formula for the successive subspace correction is defined by

$$u_{k+d/D} = u_{k+(d-1)/D} + M_d(b - A u_{k+(d-1)/D}), \quad d = 1, 2, \cdots, D. \tag{2.80}$$

Apparently, the parallel subspace correction is in the form of the additive Schwarz method, and the successive subspace correction is in the form of multiplicative Schwarz method.

Let us summarise all the above decomposition type methods here. Some of them are equivalent in terms of final formulations, the main difference is the idea behind the decomposition of the system. The subspace correction methods consider the decomposition of the finite element space, the block preconditioners are based on the partitioning of the set of indices of the system, while the Schwarz domain decomposition with matching grids divides the domain into subdomains usually by geometry. The equivalence of these methods are

- parallel subspace correction $\Longleftrightarrow$ block Jacobi $\Longleftrightarrow$ additive Schwarz
- successive subspace correction $\Longleftrightarrow$ block Gauss-Seidel $\Longleftrightarrow$ multiplicative Schwarz.

Though these methods are algebraically equivalent, each of them has its advantage. The subspace correction methods usually appear in the context of multigrid for nested cases since a coarse grid is definitely a subspace of the fine grid. In [117, 116], subspace correction is used as a mathematical tool to understand the convergence of multigrid methods. The idea of block preconditioners aids in parallel implementation of the domain decomposition methods. The Schwarz domain decomposition methods are normally viewed as the generalisation of the block preconditioners since if the grids are matching, they are algebraically equivalent, and Schwarz methods can solve more complicated problems of non-matching grids.

Chapter 3

Multigrid method

Multigrid is a multilevel linear algebra solver with two main features. It employs mesh hierarchies to remove smooth components of error, and for elliptic type PDEs, it can have $O(n)$ algorithmic complexity, where n is the problem size. Detailed descriptions of multigrid can be found in several classical texts, e.g. [27, 108, 54].

In this chapter, we give a brief introduction to multigrid methods. Three components of multigrid involve the smoother, the mesh hierarchy, and the transfer operators. There are two types of multigrid methods: geometric multigrid (GMG) and algebraic multigrid (AMG). An outline of this chapter is given in the following. An explanation of the smoothing property is provided in section 3.1 which includes several numerical examples. Mesh hierarchies used in multigrid are illustrated in section 3.2. How to construct the transfer operators in the geometric multigrid case is discussed in section 3.3 with the exact formulations given. We present a literature review of the convergence theories of multigrid in section 3.5. Algebraic multigrid is also introduced here including the classical AMG in section 3.6.1 and the smoothed aggregation AMG in section 3.6.2. A numerical example is presented in section 3.7 to validate the $O(n)$ algorithmic complexity of the multigrid preconditioned conjugate gradient solver.

3.1 Smoothing property

The smoothing property is an intrinsic property of iterative solvers, which is the essential part in multigrid. What is a smoother? A smoother is an iterative method that has the capability of eliminating the high frequency components of error. Suppose we are solving $Au = b$, where $A \in \mathbb{R}^{n\times n}$. We can decompose the+ error obtained by some smoother into two parts, smooth error containing low frequency modes and oscillatory error containing high frequency modes. Let $\{\lambda_i, w_i\}$ be the eigenvalues and orthonormal eigenvectors of the matrix A or the iterative

matrix T (eq. (2.25)). The initial error e_0 can be represented as the sum of eigenvectors of A

$$e_0 = \sum_{i=1}^{n} c_i w_i, \tag{3.1}$$

where $c_i \in \mathbb{R}$ is the coefficient giving the contribution of each mode to the error. The error obtained after k iterations of the iterative method can then be written by

$$e_k = \sum_{i=1}^{n} c_i \lambda_i^k w_i. \tag{3.2}$$

It shows that the error produced by the iterative method is made up of different eigenvectors of the matrix A. In particular, if applying finite element to solve the homogeneous Poisson equation on the unit interval, these eigenvectors of the produced finite element system become the Fourier modes [27], namely

$$w_{i,j} = \sin\left(\frac{ij\pi}{n}\right), \tag{3.3}$$

the j-th component of the i-th eigenvector of A. Therefore, the error obtained becomes a linear combination of the Fourier modes with different frequencies. It is noticed that as i increases, i.e., the frequency (wavenumber) of the Fourier mode increases, the eigenmode becomes more oscillatory. Generally there is no formal definition to distinguish high frequency and low frequency error, some texts, like [108], suggest to divide the spectrum in half.

Definition 3.1. *Suppose that the error e_k is obtained by some stationary iterative method at k-th iteration, and written in the form of eq.* (3.2)*. Then $\sum_{i=1}^{n/2} c_i \lambda_i^k v_i$ is called the smooth part of the error which corresponds to the low frequencies, while $\sum_{i=n/2}^{n} c_i \lambda_i^k v_i$ is called the oscillatory part of error which corresponds to the high frequencies.*

The so called smoothing property refers to that an iterative method can remove the high frequency oscillatory error quickly, while the smooth low frequency error typically persists. Let $e_{0,i}$, $e_{k,i}$ be the i component of the error e_0 and e_k, respectively. In chapter 2, we have seen that the basic iterative solvers reduce the error via

$$e_k = T^k e_0. \tag{3.4}$$

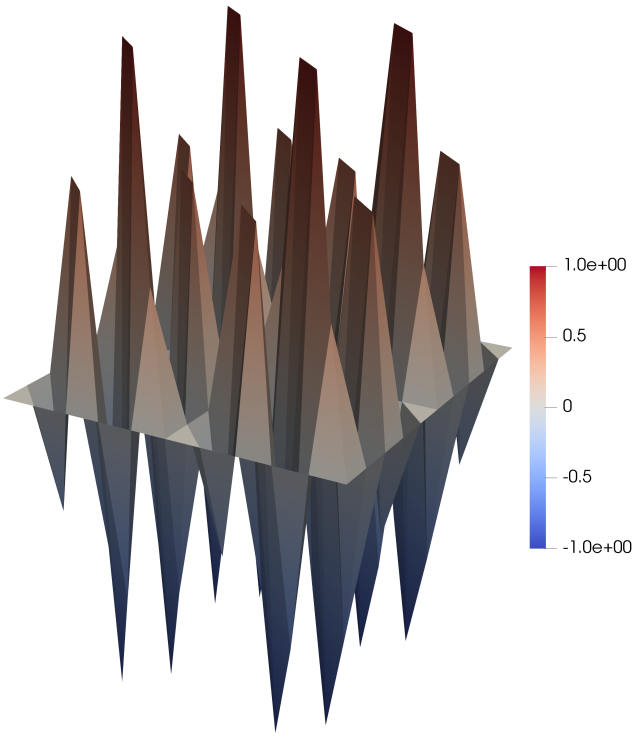


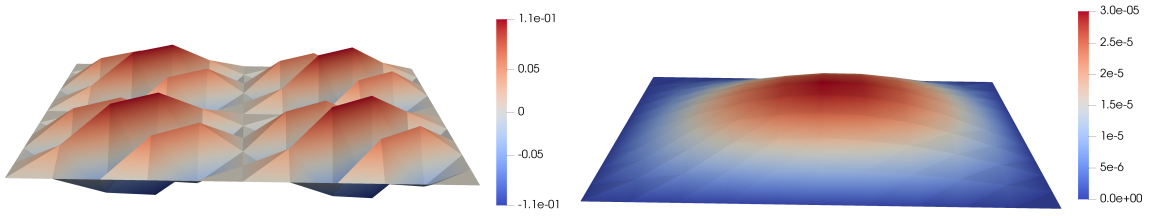
Fig. 3.1 A high frequency initial guess, $u_0 = \sin(10\pi x)(10\pi y)$.

where T is the iterative matrix. Quantitatively, the smoothing property stands for that after the smoothing process, $|e_{k,i}| \ll |e_{0,i}|$ for the high frequency parts, i.e., $n/2 \leq i \leq n$, namely high frequency error is largely removed.

Here we numerically show that many iterative methods introduced in chapter 2 possess the smoothing property. Let the domain tested here be the unit square $\Omega = (0,1)^2$. A uniform mesh is used with the cell size $1/10$. The homogeneous Poisson equation is considered with $u = 0$ on $\partial\Omega$. A high frequency Fourier mode

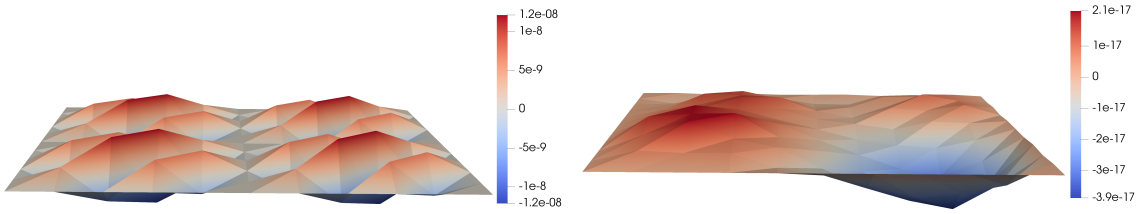
$$u_0 = \sin(10\pi x)(10\pi y) \tag{3.5}$$

is taken as the initial guess, which is shown in fig. 3.1. We first test two of the stationary methods introduced in section 2.2, the weighted Jacobi and the symmetric Gauss–Seidel methods. Figure 3.2 presents the solution after five iterations of the methods. Clearly, the high frequency parts of error are reduced dramatically by the stationary iterative solvers. Moreover, it is observed that the Gauss–Seidel is much better than the weighted Jacobi in removing the high frequency components of error. In this example, after five iterations of the Gauss–Seidel, only smooth components of error are left. It is well known that the stationary iterative solvers possess the smoothing property [54]. The best way to understand



(a) Weighted Jacobi (b) Symmetric Gauss–Seidel

Fig. 3.2 Error obtained after five iterations of (a) the weighted Jacobi and (b) the symmetric Gauss–Seidel methods.



(a) Jacobi preconditioned Chebyshev (b) Jacobi preconditioned conjugate gradient

Fig. 3.3 Error obtained by five iterations of (a) the Jacobi preconditioned Chebyshev method and (b) the Jacobi preconditioned conjugate gradient method.

the smoothing property of these smoothers is via the local Fourier analysis, which gives the insights into the decay of the high frequency modes in the error. The local Fourier analysis has been well studied for the weighted Jacobi method, see [54, 114], especially, [100] for the analysis on finite element systems.

On the other hand, Krylov subspace solvers can also serve as smoothers. The Chebyshev method discussed in section 2.5 and the conjugate gradient method discussed in section 2.3 can remove both high and low frequency error at the same time. A Jacobi preconditioner is applied to both methods. To examine the performance, the homogeneous Poisson equation is solved again with the same high frequency initial guess on the uniform unit square mesh. We show the same solution obtained by five iterations of the Krylov subspace methods in fig. 3.3. The smoothing of Jacobi preconditioned Chebyshev method is similar to the weighted Jacobi method, but the convergence is much faster. The Jacobi preconditioned conjugate gradient method can also damp out the high frequency error. In this thesis, the symmetric Gauss-Seidel and the Jacobi preconditioned Chebyshev method are employed as smoothers for multigrid.

3.2 Mesh hierarchies

In previous section, smoothing property is seen as an advantage of a smoother. However, smoothers can also suffer from the smoothing property since it is hard for them to remove the smooth components of error. Multigrid is created to deal with the remaining smooth error.

To get rid of the remaining smooth error, mesh hierarchies are provided which consist of a set of grids from fine to coarse. Multigrid exploits the coarse grid property: a smooth error on the fine grid can be well approximated on a coarse grid, and on the coarse grid, it turns out to be oscillatory, which is easy to be removed by smoother, and the coarse grid procedure is substantially less expensive. Therefore, the essence of multigrid is to correct the smooth components of error on the coarse grid.

Some notations used in constructing multigrid are described here. We use the index l to denote the level of the mesh hierarchies, and suppose there are totally L levels. The grid on level l is denoted by Ω_l with $l = 1$ being the finest grid and $l = L$ being the coarsest. The larger the level of l is, the coarser the grid turns out. Let V_l be the corresponding finite element space on the level Ω_l. The size of the problem on level l is n_l, i.e., $\dim(V_l) = n_l$. It is noted that the original multigrid, both theory and implementation, was constructed with nested meshes, which refer to that the nodes on the coarse grid are also present on the fine grid, i.e., $V_{l+1} \subset V_l$. General multigrid procedure can now be extended to the non-nested meshes, $V_{l+1} \not\subset V_l$. In this research, we consider most problems of using non-nested mesh levels, which is the case happened in many engineering simulations.

There are two main types of multigrid: geometric multigrid (GMG) and algebraic multigrid (AMG). They both share the same algorithm, and the main difference is the way of generating the mesh hierarchies. In GMG, the mesh hierarchies are given geometrically, namely, we have the detailed geometric meshes for all coarse grids. For AMG, the mesh hierarchies are generated algebraically, namely, the coarse grids are constructed algebraically from the fine grid system. From section 3.3 to section 3.5, we give an overview of GMG, and two AMG algorithms are introduced in section 3.6.

3.3 Transfer operators

Transfer operators transfer the solution function between grids. They include both the prolongation and the restriction operators. The prolongation operator maps functions from the coarse grid to the fine grid,

$$P_l : V_{l+1} \to V_l, \tag{3.6}$$

which is a linear projection of size of $n_{l+1} \times n_l$. The restriction operator projects the fine grid functions to the coarse grid,

$$R_l : V_l \to V_{l+1}, \tag{3.7}$$

which is a linear map of size $n_l \times n_{l+1}$. In the Galerkin approximation [27, 19], the restriction operator is set to be the transpose of the prolongation,

$$R_l = P_l^T. \tag{3.8}$$

The construction of the transfer operators is not unique. The choice of constructing the transfer operators has a significant impact on the rate of convergence of multigrid [58].

In the finite difference scheme, uniform and nested meshes are used, and the prolongation operator can easily be assembled based on the geometries of the mesh levels [108]. For the finite element, especially for complicated engineering applications, non-nested and unstructured meshes can be employed. The prolongation is constructed by the consistency of nodal values between grids. It is noted that multigrid on non-nested meshes is well discussed in [106]. Suppose the levels are non-nested $V_{l+1} \not\subset V_l$. Let functions u_l and u_{l+1} be the same solution defined on the fine and the coarse grids respectively, and φ^l and φ^{l+1} be the basis functions on the fine and the coarse grids respectively. We have two corresponding functions on the fine and the coarse grids with

$$\begin{aligned} u_l(x) &= \sum_{i=1}^{n_l} a_i^l \varphi_i^l(x) \\ u_{l+1}(x) &= \sum_{j=1}^{n_{l+1}} a_j^{l+1} \varphi_j^{l+1}(x), \end{aligned} \tag{3.9}$$

where a_i^l and a_j^{l-1} are DOFs on each grid with $a_i^l = u_l(x_i)$. At any point x_i, the two functions should give the same value,

$$u_{l+1}(x_i) = u_l(x_i) = a_i^l, \tag{3.10}$$

thus

$$a_i^l = \sum_{j=1}^{n_{l+1}} a_j^{l+1} \varphi_j^{l+1}(x_i). \tag{3.11}$$

The prolongation, mapping the DOFs on the coarse grid a_i^{l+1} to the DOFs on the fine grid a_i^l, can be written as

$$P \begin{bmatrix} a_1^{l+1} \\ a_2^{l+1} \\ \vdots \\ a_{n_{l+1}}^{l+1} \end{bmatrix} = \begin{bmatrix} a_1^l \\ a_2^l \\ \vdots \\ a_{n_l}^l \end{bmatrix}. \tag{3.12}$$

Note that the i-th row of the mapping should correspond to eq. (3.10), which gives

$$\begin{aligned} \sum_{j=1}^{n_{l+1}} P_{ij} a_j^{l+1} &= a_i^l \\ \sum_{j=1}^{n_{l+1}} P_{ij} a_j^{l+1} &= \sum_{j=1}^{n_{l+1}} a_j^{l+1} \varphi_j^{l+1}(x_i), \end{aligned} \tag{3.13}$$

and by comparing the LHS and RHS, it follows that

$$P_{ij} = \varphi_j^{l+1}(x_i). \tag{3.14}$$

The formulation of this prolongation works for non-nested case and i, j refer to the nodes/DOF indices on the fine and coarse grids, respectively.

It is found that the construction of the prolongation operator is not cheap, it takes $O(n_l \log n_{l-1})$ operation cost [97]. A search tree can be carried out to track the element on coarse grid containing x_i, which helps improve the efficiency of constructing the prolongation. Denote x_i^l as the fine grid node and x_j^{l+1} as the coarse grid node. For any point x_i^l on the fine grid, if there exists some node j on the coarse grid such that $x_i^l = x_j^{l+1}$, then $P_{ij} = 1$. If there is no such node j on the coarse grid, we track the element τ^{l+1} on the coarse grid such that $x_i^l \in \tau^{l+1}$, and then for $j \in \tau^{l+1}$, we have $P_{ij} = \varphi_j^{l+1}(x_i^l)$. The construction of the prolongation for non-nested meshes is illustrated in algorithm 3.1.

Algorithm 3.1 Construction of the prolongation operator

1: **for** $i = 1, 2, \cdots, n_l$ **do**
2: **if** x_i^l is also a nodal point on the coarse grid with $x_i^l = x_j^{l+1}$ **then**
3: $[P_l]_{ij} = 1$
4: **else** x_i^l is inside some element on the coarse grid with $x_i^l \in \tau^{l+1}$
5: **for** $j \in \tau^{l+1}$ **do**
6: $[P_l]_{ij} = \varphi_j^{l+1}(x_i^l)$

Example 3.1. *Finally we give an example for construing the prolongation on a 1D line segment. Suppose there are 11 nodes with $x_0, x_1, \cdots, x_{10}$ on the fine grid, and there are 5*

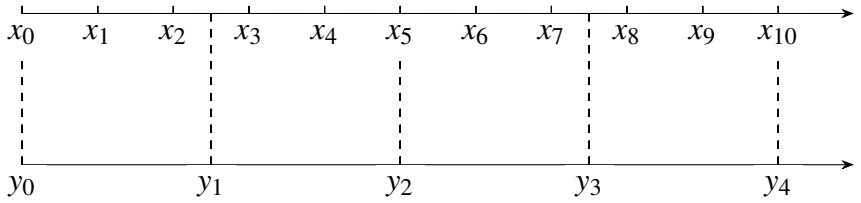


Fig. 3.4 Fine and coarse grids of a line segment mesh, in which x_i are nodes on the fine grid and y_j are nodes on the coarse grid.

nodes with $y_0, y_1, \cdots, y_4$ on the coarse grid. The meshes are shown in fig. 3.4. The resulting prolongation operator is given by

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \varphi_0(x_1) & \varphi_1(x_1) & 0 & 0 & 0 \\ \varphi_0(x_2) & \varphi_1(x_2) & 0 & 0 & 0 \\ 0 & \varphi_1(x_3) & \varphi_2(x_3) & 0 & 0 \\ 0 & \varphi_1(x_4) & \varphi_2(x_4) & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \varphi_2(x_6) & \varphi_3(x_6) & 0 \\ 0 & 0 & \varphi_2(x_7) & \varphi_3(x_7) & 0 \\ 0 & 0 & 0 & \varphi_3(x_8) & \varphi_4(x_8) \\ 0 & 0 & 0 & \varphi_3(x_9) & \varphi_4(x_9) \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.15}$$

In terms of implementation, the prolongation operator can be constructed by create_transfer_matrix operation in PETScDMCollection, which is implemented in the software DOLFIN of the FEniCS [80]. And, we use this implementation in all our numerical examples.

3.4 Multigrid cycles

After we obtain the transfer operators, we can present the algorithm of multigrid. Let A_l be the linear system to solve on Ω_l. The coarse grid operator is now defined by the so-called Galerkin projection [27, 54],

$$A_{l+1} = R_l A_l P_l. \tag{3.16}$$

The Galerkin projection can be shown to be well suited in multigrid scheme in the later context. A smoother S is applied on the level Ω_l ($l \neq L$), the application of which is denoted

by

$$u_l \leftarrow S^{\nu}(A_l, b_l, u_l), \tag{3.17}$$

where ν is the number of iterations applied. The smoothers for multigrid used here, as discussed in section 3.1, are normally the symmetric Gauss-Seidel and the Jacobi preconditioned Chebyshev methods. Multigrid exploits the coarse grid correction, namely, it solves the residual equation on the coarse grid. In order to view multigrid clearly, we often write multigrid as a cycle. If the Galerkin finite element discretisation carried out on the finest grid Ω_1 generates the linear system

$$A_1 u_1 = b_1, \tag{3.18}$$

then a V-cycle multigrid consisting of L levels is presented in algorithm 3.2 to solve the linear system. The algorithm is called a V-cycle multigrid since it looks like the letter 'V',

Algorithm 3.2 Multigrid V-cycle of L levels to solve $A_1 u_1 = b_1$.

Starting with $l = 1$
procedure $u_l = \text{VCYCLE}(A_l, b_l, u_l, l, \nu)$
  if $l \neq L$ **then**
    Pre-smoothing $u_l \leftarrow S^{\nu}(A_l, b_l, u_l)$
    Calculating the residual $r_l = b_l - A_l u_l$
    Coarse grid construction $A_{l+1} = P_l^T A_l P_l$, $b_{l+1} = P_l^T r_l$ and $u_{l+1} = P_l^T u_l$
    Coarse grid correction $u_{l+1} = \text{Vcycle}(A_{l+1}, b_{l+1}, u_{l+1}, l+1, \nu)$
    Updating current solution $u_l \leftarrow u_l + P_l u_{l+1}$
    Post-smoothing $u_l \leftarrow S^{\nu}(A_l, b_l, u_l)$
  else
    Direct solver on the coarsest grid $u_L = A_L^{-1} b_L$

shown in fig. 3.5. There are other multigrid cycles like the W-cycle and the full multigrid scheme, which are shown in fig. 3.6. These cycle are well illustrated in [27, 108].

We now show the validation of the Galerkin projection. For convenience, we look into a two–level case with the fine level l and the coarse level $l+1$. As discussed in section 2.3, we have the variational form for the linear system $Au = b$ defined as

$$F(u) = \frac{1}{2} u^T A u - u^T b. \tag{3.19}$$

If u^* solves the linear system, then we have

$$u^* = \arg\min_{v \in V} F(v). \tag{3.20}$$

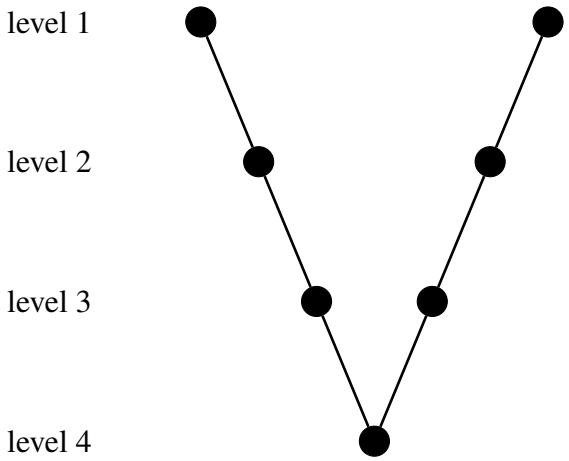


Fig. 3.5 A four–level multigrid V-cycle.

For a two-level multigrid V-cycle, let $v_l \in V_l$ and $w_{l+1} \in V_{l+1}$ be functions on the two levels, the variational forms of systems, F_l for the fine grid and F_{l+1} for the coarse grid, are formulated by

$$\begin{aligned} F_l(v_l) &= \frac{1}{2} {v_l}^T A_l v_l - {v_l}^T b_l, \\ F_{l+1}(w_{l+1}) &= \frac{1}{2} {w_{l+1}}^T A_{l+1} w_{l+1} - {w_{l+1}}^T b_{l+1} \\ &= \frac{1}{2} {w_{l+1}}^T (R_l A_l P_l) w_{l+1} - {w_{l+1}}^T [R_l (b_l - A_l v_l)]. \end{aligned} \tag{3.21}$$

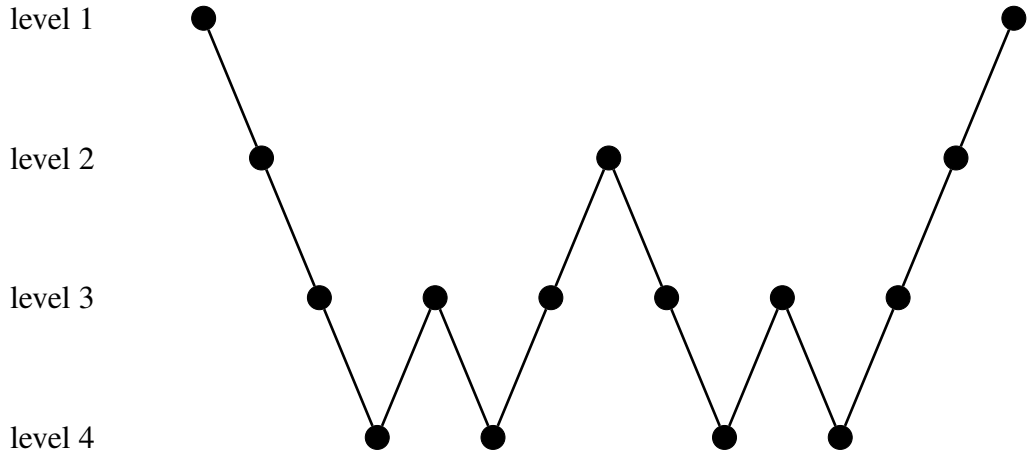
The following theorem shows the validity of the Galerkin projection on a two-level system.

Theorem 3.1. *Suppose that there are two levels, fine and coarse. Let v_l be some approximation to the fine grid system, and u_{l+1} be the exact solution to the coarse grid system. Then the following holds*

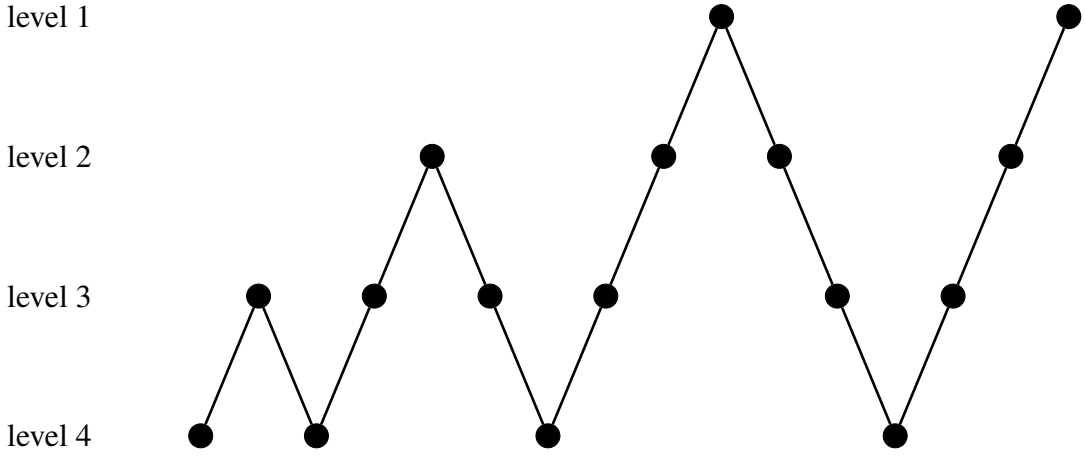
$$u_{l+1} = \underset{w_{l+1} \in V_{l+1}}{\arg\min}\ F_l(v_l + P w_{l+1}), \tag{3.22}$$

where P is the prolongation constructed in eq. (3.14).

The proof can be found in [54]. The theorem supports that after a two-level v-cycle, with the solution prolongated from the coarse grid exact solution, the variational form on the fine grid can be minimised. Therefore, the coarse grid correction used in multigrid provides an optimal solution to the linear system on the fine grid.



(a) A four–level multigrid W-cycle



(b) A four–level full multigrid scheme

Fig. 3.6 Two multigrid cycles, (a) W-cycle and (b) full multigrid scheme.

3.5 Convergence results

We give a literature review of the convergence of multigrid methods here. The idea of multigrid was proposed by Fedorenko [41, 42], who formulated a two-level method for solving the Poisson equation in a unit square and discovered its $O(n)$ algorithmic complexity. The pioneering work by Brandt [23] claimed the new method and gave the name 'multigrid' in the mid-seventies. There were several independent approaches to prove the convergence of multigrid including Hackbusch [52], Nicolaides [86], and Bank and Dupont [13]. Some comprehensive proofs on multigrid with non-nested mesh levels have been studied in [20, 98, 16]. A general framework breaks down the analysis into two two separate parts: the smoothing property and the approximation property. The formal proof of convergence starts with establishing these properties in terms of the mesh dependent norm,

$$|||u|||_{s,l} := (u_l, A^s u_l)^{1/2} := \sqrt{u_l^T A^s u_l}. \tag{3.23}$$

The smoothing property is usually based on a straightforward relaxation method, e.g. the weighted Jacobi method. The approximation property is based on the *a priori* estimates, discussed in theorem 2.1, which can be extended to the mesh dependent norm. Combining this two properties, convergence result for multigrid is obtained. The famous convergence theorem on multigrid V-cycle is shown in the following, which is in the energy norm.

Theorem 3.2. *Suppose $u_k = Vcycle(A, b, u_{k-1}, L, \nu)$, and u^* is the exact solution to $Au = b$, then*

$$\|u_k - u^*\|_E \leq \frac{c}{\nu + c} \|u_0 - u^*\|_E \tag{3.24}$$

where c is some constant independent of the level L, and ν is the number of iterations of the smoother applied in each pre- and post- smoothing.

The detailed proof of theorem 3.2 can be found in [26]. This convergence theorem is based on the geometric multigrid. Two main features of the convergence of the geometric multigrid can be generalised here: (i) The convergence of multigrid can work for both structured and unstructured meshes, and also for the non-nested meshes between levels; (ii) The convergence of the geometric multigrid is independent of the number of levels used. Because of the clear and simple convergence results, GMG is commonly favoured over AMG for which convergence may depend on number of levels.

An important property of multigrid is its computational cost. It is well known that multigrid algorithm can have $O(n)$ algorithmic complexity for solving elliptic type PDEs [54, 108], which means that the computational time increases linearly with the size of the problem. In fact, this result holds firmly with the full multigrid scheme, shown in fig. 3.6b.

Let n_l be the number of DOFs on the level l, if the smoother on level l has work cost of $O(n_l)$, then the number of operations W_l on level l of the full multigrid scheme is bounded by

$$W_l \leq Cn_l, \tag{3.25}$$

where C is a constant. For elliptic type PDEs on uniform meshes, the number of multigrid cycles for solving the linear system is bounded uniformly, i.e., independent of number of levels, then it follows that the full multigrid scheme takes $O(n)$ operations to solve the system. Moreover, when using multigrid as a preconditioner, the condition number of the preconditioner is bounded, i.e. $\kappa(MG(A)) \sim O(1)$, then the work cost of multigrid precondtioned conjugate gradient method is also $O(n)$.

3.6 Algebraic multigrid

The advantage of geometric multigrid (GMG) is its fast convergence, which is independent of number of levels. However, GMG suffers from the requirement of the geometric hierarchy of meshes. For complicated engineering simulations, creating coarse grids can be time-consuming, and many times, geometric mesh hierarchies are not feasible. Algebraic multigrid (AMG) methods were designed in an attempt to address this limitation of GMG. AMG methods construct the coarse grids by the algebraic information of the fine grid. There are a variety of AMG methods that have been developed, see the review papers [105, 118]. We give a brief introduction of two main AMG methods: the classical AMG and the smoothed aggregation AMG (SA-AMG).

3.6.1 Classical AMG

The classical AMG regards the coarse level nodes as a subset of the fine level ones. The coarsening strategy for the classical AMG is called the C/F splitting, which is based upon the graph of the current grid and the idea of algebraic smoothness. The method was introduced in [94], and its details can be found in texts [27, 108].

It is noted that the transfer operators for the classical AMG are always constructed after a suitable C/F splitting is obtained. Let the set of DOFs on the fine grid be $N = \{1, 2, \cdots, n\}$. The C/F splitting splits this set to two disjoint subsets by

$$N = C \cup F \quad \text{and} \quad C \cap F = \emptyset, \tag{3.26}$$

where the set C contains those nodes selected to serves as coarse grid nodes and the set F is the set of nodes remaining on the fine grid. We first give the terminology of the strong coupling.

Definition 3.2. *Let a_{ij} be the i,j entry of the matrix A. A node i is said to be strongly (negatively) coupled to another node j if*

$$-a_{ij} \geq \varepsilon \max_{a_{ik}<0} |a_{ik}|, \tag{3.27}$$

where $\varepsilon \in (0,1)$ is a fixed number called the coarsening factor.

The splitting scheme is then a selection algorithm that selects those nodes on the fine grid with the largest measure of strong couplings to be those on the coarse grid. In order to give the measure of strong couplings for each node, we define several relevant sets here.

Definition 3.3. *The neighbourhood set of node i is*

$$N_i = \{j \in N : \ j \neq i, a_{ij} \neq 0\}. \tag{3.28}$$

The set of all strongly couplings of node i is

$$S_i = \{j \in N_i : \ i \text{ is strongly coupled to } j\}. \tag{3.29}$$

The strongly transpose coupling set of node i consists of all nodes j which are strongly coupled to i, defined by

$$S_i^T = \{j \in N_i : \ i \in S_j\}. \tag{3.30}$$

The measure of strong couplings for each node i is then given by

$$\lambda_i = |S_i^T| + 2|S_i^T \cap F|, \tag{3.31}$$

where $|\cdot|$ stands for the size of the set. The C/F spitting scheme with this measure [27] is described in fig. 3.7.

After a suitable coarse grid is obtained by C/F splitting, the prolongation operator is then constructed. The rule of constructing the prolongation operator is the wish to reduce the algebraic smooth error on the coarse grid. Let e_f and e_c be the error on the fine and the coarse grids respectively, a direct prolongation operator P can be formulated as

$$[e_f]_i = [Pe_c]_i = \begin{cases} [e_c]_i & \text{if } i \in C, \\ \sum\limits_{k \in K_i} \omega_{ik} [e_c]_k & \text{if } i \in F, \end{cases} \tag{3.32}$$

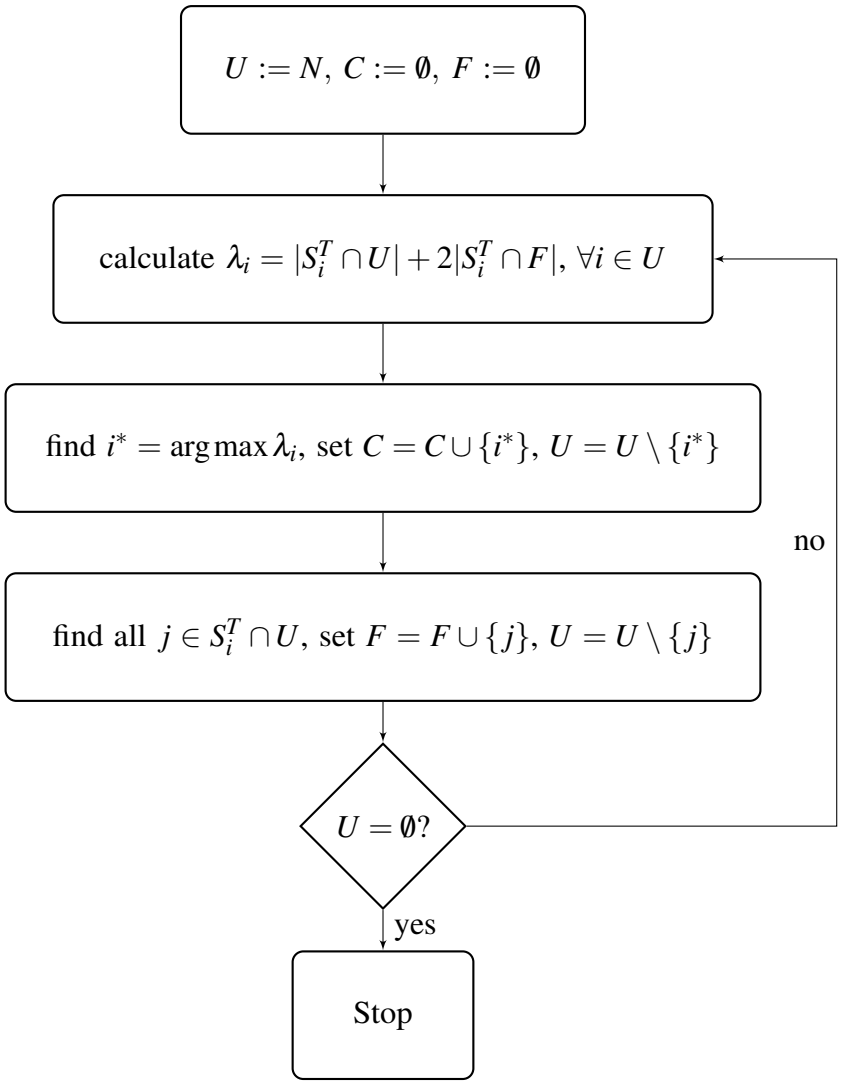


Fig. 3.7 C/F splitting algorithm for the classical AMG, in which U is the set of undecided nodes, C is the set of coarse grid nodes, and F is the set of fine gird nodes that are not on the coarse grid.

where K_i is the interpolatory set of node i with $K_i \subset C$, and ω_{ik} is some constant depending on K_i used to remove the smooth error. The so called direct prolongation chooses the set of interpolatory nodes K_i by

$$K_i = C_i^s := C \cap S_i. \tag{3.33}$$

With this interpolatory set, the corresponding weight in eq. (3.32) is given by

$$\omega_{ik} = -\alpha_i \frac{a_{ik}}{a_{ii}} \quad \text{with} \quad \alpha_i = \frac{\sum\limits_{j \in N_i} a_{ij}}{\sum\limits_{k \in K_i} a_{ik}}. \tag{3.34}$$

Basically, theories of the classical AMG [94] are restricted to the case that A is a symmetric positive definite M-matrix which means the off–diagonal entries are less than and equal to zero and the real parts of eigenvalues are nonnegative. It has been seen that the classical AMG with the direct prolongation is good at solving the scalar elliptic PDEs in which largest off-diagonal entries are usually negative. The dominant negative components at off-diagonal entries fit well with the idea of strongly negative coupling. However, the main disadvantage of classical AMG is the lack of convergence theories in terms of the number of levels. There is a view in the field that even if a two-level classical AMG works, it does not necessarily mean that multilevel AMG can also converge in the same way. Experiments have shown that the convergence rate can decay if more levels are used [105].

3.6.2 Smoothed aggregation AMG

In the mid 90's, there is a strong need for a powerful algebraic multigrid since classical AMG is too much dependent on strongly coupling system, which has too many constraints. The theory for classical AMG is based on the two-level estimates, which does not include the error bounds independent of the number of levels. Smoothed aggregation AMG (SA-AMG) was created by Vaněk [110] to serve as a fast solver to solve the linear elasticity equation. It has been shown that the smoothed aggregation AMG can be applied to extremely large complicated applications in engineering [93].

The coarsening strategy in the smoothed aggregation AMG relies on the creation of aggregates of the domain. An aggregate refers to some subdomain/subset on the fine grid and it gives rise to a DOF on the coarse grid. By doing this, the main feature of the smoothed aggregation AMG is that it tries to construct a reasonable set of coarse grid basis functions.

Let V_l be the fine grid finite element space with its basis set $\{\varphi_i^l\}_{i=1}^{n_l}$, and V_{l+1} be the coarse grid space with basis set $\{\varphi_i^{l+1}\}_{i=1}^{n_{l+1}}$. The restriction operator P_l^T based on the

aggregation maps the basis function from fine to coarse

$$\begin{bmatrix} \varphi_1^{l+1} \\ \varphi_2^{l+1} \\ \vdots \\ \varphi_{n_{l+1}}^{l+1} \end{bmatrix} = P_l^T \begin{bmatrix} \varphi_1^{l} \\ \varphi_2^{l} \\ \vdots \\ \varphi_{n_l}^{l} \end{bmatrix}. \tag{3.35}$$

Moreover, the smoothed aggregation AMG aims to minimise the energy of the coarse grid basis functions, which is a measure in the energy norm defined by

$$\|u_l\|_E = u_l^T A_l u_l. \tag{3.36}$$

With the coarse grid basis function of minimised energy, the coarse grid system is algebraically equivalent to the finite element system of the coarse grid constructed by these basis functions.

The algorithm to construct aggregates is not trivial, we give a brief idea here. It is noted that the coarsening algorithms for different element types and orders are slightly different. The selection scheme is still based on the graph of the current system. The strong coupling in the smoothed aggregation AMG is defined in the following.

Definition 3.4. *The node i is strongly coupled to node j if*

$$|a_{ij}| \leq \varepsilon \sqrt{a_{ii} a_{jj}}, \tag{3.37}$$

where $\varepsilon \in [0,1)$ is still the coarsening parameter.

Note the different definitions of strong coupling in classical AMG and smoothed aggregation AMG. This definition in smoothed aggregation AMG allows it to work well for matrices containing significant positive off-diagonal entries. Let the linear system on the fine level be A_l of size n_l. The coarsening procedure intends to find the disjoint subsets $\{\mathscr{A}_j^l\}_{j=1}^{n_{l+1}}$ of the set $\{1,2,\cdots,n_l\}$, and each subset $\mathscr{A}_j^l$ contains DOFs on the fine grid which make up one aggregate. There are three steps to make a standard aggregation, which is shown in the following.

• Initialisation: Let the undecided set be $U = \{1,2,\cdots,n_l\}$, and $j = 0$.

• Step 1: For all $i \in U$, find the strongly-coupled neighbourhood S_i. If $S_i \subset U$, then set $j = j+1$, $\mathscr{A}_j^l = S_i$, and $U = U \setminus \mathscr{A}_j^l$.

• Step 2: Let $\tilde{\mathscr{A}}_k^l = \mathscr{A}_k^l$ for $K = 1,2,\cdots,j$. For all nodes remaining $i \in U$, if there exists k such that $S_i \cap \tilde{\mathscr{A}}_k^l = \emptyset$, then set $\mathscr{A}_k^l = \mathscr{A}_k^l \cup \{i\}$, and $U = U \setminus \{i\}$. Repeat this procedure until no such i exists.

• Step 3: If there is still some $i \in U$, then set $j = j+1$, $\mathscr{A}_j^l = U \cap S_i$, and $U = U \setminus \{i\}$. Repeat this procedure until all nodes are assigned to some aggregate, i.e., $U = \emptyset$.
After all aggregates $\mathscr{A}_j^l$ are found, then these aggregates make up the coarse grid DOFs.

A tentative prolongation $\tilde{P}_l$ is first constructed with

$$\tilde{P}_l[i,j] = \begin{cases} \alpha_i & \text{if } i \in \mathscr{A}_j^l \\ 0 & \text{otherwise} \end{cases}, \tag{3.38}$$

where α_i equals one for scalar P_1 problems and some constant for higher order or vector-valued problems. For example, the tentative prolongation for a 1D Laplace equation has the structure

$$\tilde{P} = \begin{bmatrix} \alpha_1 & & & \\ \alpha_1 & & & \\ \alpha_1 & & & \\ & \alpha_2 & & \\ & \alpha_2 & & \\ & \alpha_2 & & \\ & & \vdots & \\ & & & \alpha_n \\ & & & \alpha_n \\ & & & \alpha_n \end{bmatrix}. \tag{3.39}$$

It can be seen that the tentative prolongation only connects the node on the fine grid to its corresponding aggregate of the DOF on the coarse grid. However, it is found that the tentative prolongation is weak in terms of minimising the energy of coarse grid basis functions in eq. (3.36) [110, 109]. Thus, a smoothed prolongation is introduced by applying a simple weighted Jacobi smoother to the tentative prolongation via

$$P_l = S_l \tilde{P}_l = (I - \omega D_l^{-1} A_l)\tilde{P}_l \tag{3.40}$$

where D_l is the diagonal matrix of A_l, and ω is defined as $\omega = 4/(3\lambda_{\max}(D_l^{-1}A_l))$. This step preserves the properties of the tentative prolongation while decreasing the energy of the coarse grid basis functions. A more detailed description of the smoothed aggregation AMG setting can be found in [110]. We make a remark here that, in this thesis, the largest eigenvalues used in constructing the smoothed prolongation is approximated by the Krylov Schur method [104, 122, 59], which is implemented in the SLEPc and PETSc [11, 10, 12].

In order to make the approximation accurate, a tolerance of the residual, i.e. $\|Au - \lambda u\|_2$ is set as 10^{-3}.

Finally, the coarse grid operator by the smoothed aggregation AMG is defined as

$$A_{l+1} = (S_l \tilde{P}_l)^T A_l (S_l \tilde{P}_l) = \tilde{P}_l^T \left(I - \frac{4}{3\lambda} D_l^{-1} A_l \right)^T A_l \left(I - \frac{4}{3\lambda} D_l^{-1} A_l \right) \tilde{P}_l, \tag{3.41}$$

where λ is the maximum eigenvalue of $D_l^{-1} A_l$. This setting is the so-called Ritz-Galerkin AMG, and we focus on this setting since it is the most commonly used in engineering and satisfies the Galerkin projection. However it has been found that if we use the smoothed prolongation, the coarse grid systems become less sparse, which may cause much higher computational costs on the coarse levels. Non-smoothed aggregation AMG and half-smoothed aggregation AMG are also employed to balance the computational time and the convergence rate. Some research have been carried out in this direction like the Petrov-Galerkin AMG in [50] where they use the coarse grid system

$$A_{l+1} = \tilde{P}_l^T A_l (S \tilde{P}_l). \tag{3.42}$$

Some convergence theories are studied for the Ritz-Galerkin case in [109]. Smoothed-aggregation AMG is still like a 'black box' in multigrid, there is a lack of convergence studies, while numerically the smoothed aggregation AMG method works well for a large number of problems. It is emphasised here that the near–null space plays an essential role in aggregation based AMG especially for vector-valued problems. For scalar-valued problems, the dimension of the near–null space is one, while for elasticity problems, it becomes three (two constants and one linear function) for 2D, and six (three constants and three linear functions) for 3D. In particular, for 3D linear elasticity, the near–null space corresponds to the six rigid body modes including translations and rotations. The detail implementation of the near–null space in the smoothed aggregation AMG can be found in [46].

3.7 Numerical examples

In this section, we apply the multigrid preconditioned conjugate gradient method to verify its performance. The problem tested here is the homogeneous Poisson equation on a unit square with the homogeneous boundary condition. The unit square mesh hierarchies are all unstructured and non-nested. Finite element method is carried out with P_1 elements. All the geometric multigrid (GMG), the classical AMG, and the smoothed aggregation AMG are tested as the preconditioners for the conjugate gradient. Table 3.1 lists the number

of the preconditioned conjugate gradient method needed to reduce the relative residual to 10^{-10}. Clearly, for all tested multigrid preconditioners, the numbers of iterations needed are kept almost the same for different sizes of problems, which supports the $O(n)$ algorithm complexity.

Number of DOFs	Preconditioners		
	GMG	classical AMG	smoothed aggregation AMG
157	5	7	7
585	6	7	7
2257	6	7	7
8865	6	8	7
35137	6	8	7

Table 3.1 Number of iterations of the preconditioned conjugate gradient method needed to reduce the relative residual to 10^{-10}.

Chapter 4

A local correction smoother

As shown in section 1.3, the performance of multigrid degrades dramatically with low quality meshes, possibly leading to a failure to converge. When meshing geometrically complex domains for engineering applications, clusters of poor quality cells are not uncommon. Ideally, the mesh quality would be improved at generation time, but this may be difficult for complex geometries. Moreover, mesh generation may be performed by one team, and the analysis performed by another. The delay caused by the analyst returning a mesh to the model creator for improvement can be substantial and unacceptable in a design and analysis process. Therefore, finding techniques to overcome poor solver performance in the presence of a small numbers of low quality cells is appealing in order to increase the robustness (and by extension acceptance) of multigrid methods in engineering practice.

In this chapter, we seek a way to fix geometric multigrid to restore the convergence rate. To tackle this problem, we first examine the performance of geometric multigrid on a simple mesh in presence of a small number of low quality cells, and find the reason behind the degraded performance. It is found that the smoothing property of the smoother is damaged locally in regions of low quality cells. Building on this observation, we propose a global–local combined smoother which is a two-step method including (i) a regular multigrid smoother applied over the entire domain, followed by (ii) a local correction smoother on the small regions with low quality cells. The local regions where the local correction smoother applies are tracked by setting a threshold on the normalised radius ratio.

Several numerical examples that range from illustrative to large are also presented to validate the performance of the combined smoother. In the context of geometric multigrid, we consider two cases in which low quality cells are present. In the first case, there are poor quality cells on all grids including the finest grid. Another case considers the situation when the poor quality cells only appear on the intermediate grids. The second case is appealing for complex engineering geometries in which the fine grid may be of high quality to provide

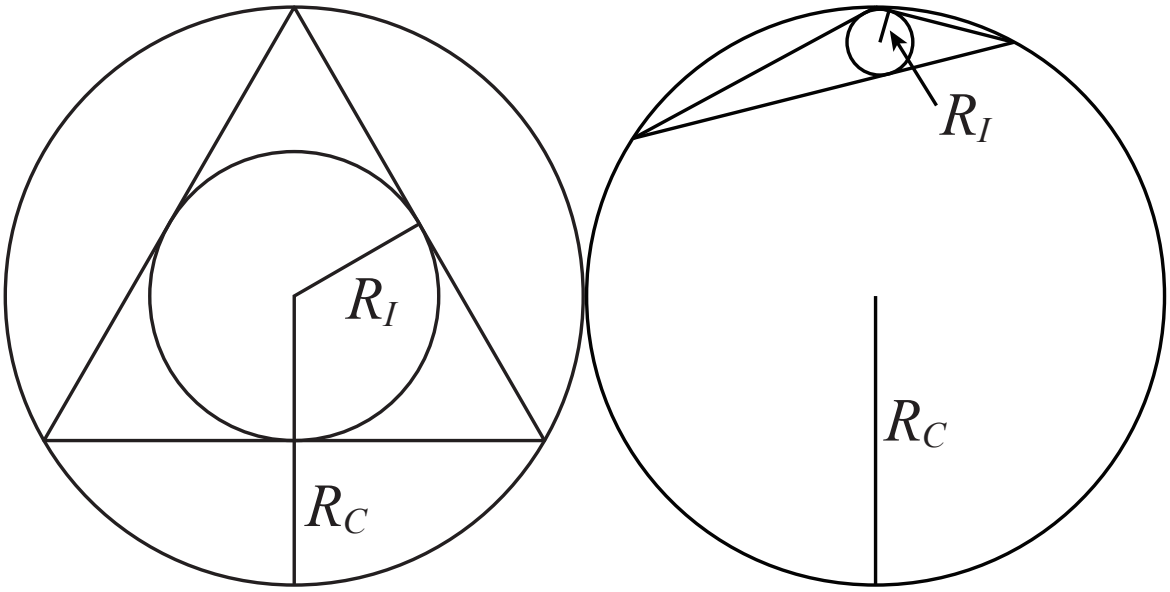
good approximation properties, but the coarse grids are just a vehicle for construing fast a solver. All tests are performed on unstructured and non-nested meshes. It is shown that the the poor convergence on low quality meshes can be restored to the reference level of using high quality meshes. Moreover, the finite element error for low quality meshes can be reduced with the use of local correction.

An outline of this chapter is as follows. The mesh quality is discussed in section 4.1, which contains the description of the radius ratio. Numerical examples are presented in section 4.2 to show the poor performance of multigrid with low quality meshes, and we also discover the local failure of the smoothing property. The local correction smoother is explained in section 4.3 with the specific algorithm given. Section 4.4 gives the exact formula of the global–local combined smoother, which is found to be equivalent to the Schwarz type domain decomposition method. Numerical experiments are shown in section 4.5 to validate the performance of the proposed smoother. Finally, we draw some concluding remarks in section 4.6.

4.1 Mesh quality measures

Finite element method requires a reasonable mesh that well represents the domain Ω. In engineering applications, complicated geometries are commonly encountered, in which case structured grids are not feasible, and it is necessary to employ unstructured grids. The topology of cells in unstructured grids is completely different, thus mesh quality plays an essential role in finite element approximation and the performance of iterative methods. There are a variety of cell quality measures for simplex cells. A detailed introduction and comparison of some common cell qualities are given in [89]. It is stated in [89] that there is no optimal cell quality measure, since each measure has its limited effective region. For example, the most classical and obvious cell quality is the minimum dihedral angle, but the low quality 'wedge-like' element usually has relatively normal size of dihedral angle. In this research, we choose to use the radius ratio which evaluates the fraction of a triangle's or tetrahedron's circumscribing sphere radius (R_C) to its inscribed sphere radius (R_I). The optimal radius ratios for the equilateral triangle and tetrahedron, which are often viewed as the 'best' elements, are 2 and 3 respectively. In order to have a simple visualisation, we normalise the radius ratio from 0 to 1 via

$$\gamma := \frac{R_I}{R_C} * \gamma^*, \tag{4.1}$$



(a) A high quality triangular element with $\gamma = 1$ (b) A low quality triangular element with $\gamma = 0.03$

Fig. 4.1 Examples of high and low quality triangular cells with normalised radius ratio.

where γ^* is the optimal radius ratio, and then the optimal cell has the measure one, while a degenerate cell has the measure zero. The radius ratio is the most commonly used mesh quality for simplex cells for its relatively larger effective regions. There is still no clear boundary of the mesh quality measure for catergoring a mesh is of high quality or low quality. In general, the radius ratio of all elements in the mesh should be as close as possible to 1, but this is not straightforward in practice. An example is made in fig. 4.1 with a pair of high and low quality triangular elements.

The mesh quality has a large effect on the finite element approximation. Some studies on the angle condition for the simple uniform structured mesh to ensure the convergence of the finite element method have been done in [9, 71]. However, these bounds only work for really simple cases, and may only be the sufficient condition for convergence (see the latest development [57]). The studies of the influence of mesh quality on the properties of the resulting FEM matrix have been carried out largely, especially the influence on the condition number, e.g., [55, 35, 36]. On the other hand, the mesh quality also has a big impact on the convergence of iterative solvers to solve linear systems generated from the finite element method. Low quality meshes can dramatically slow down the convergence of iterative solvers, which has already been observed in engineering applications, e.g. [102, 45, 65].

In practice, low quality meshes often exist in science and engineering applications. In order to tackle the issue brought by the low quality cells, much attention has been paid to improve the mesh quality in different ways. For instance, there are many geometric methods

to improving mesh quality like [68, 69, 67], in which the vertices in the local poor quality regions are moved based on the optimisation of some mesh quality functional. However, improving defects can be time costly, especially if meshes are generated by one team, and analysed by another. It is almost impossible to obtain an ideal mesh for large complex domains. In many cases, a mesh may contain mostly high quality cells except a very small number of low quality cells. The locally low quality cells can be hard to fix *a posteriori*. In particular, it is found in [36] that the standard *a priori* error estimate still remains valid on the finite element meshes with local damages, but the condition number of the resulting finite element system becomes arbitrarily large, which may degrade the performance of iterative solvers. Therefore, it is reasonable to keep working on these meshes with a small number of locally poor quality cells.

4.2 Multigrid on low quality meshes

4.2.1 Poor performance of multigrid

The performance of multigrid degrades dramatically with poor quality meshes. It is not surprising and has already been observed in engineering applications [93]. We show two examples here where convergence of multigrid decays significantly with low quality meshes.

Example I First, we use an unstructured unit square mesh $\Omega = (0,1)^2$ containing 158 nodes with one low quality region at the centre, which is shown in fig. 4.2. There is an extremely thin, and almost degenerate triangle cell in the mesh. The minimum angle of that triangle is about 1°, which leads to the normalised radius ratio of 10^{-3} in that cell. A two level multigrid V-cycle is carried out with a high quality coarse grid mesh containing 45 DOFs. The homogeneous Poisson equation

$$\begin{aligned} -\nabla^2 u &= 0 \quad \text{in } \Omega, \\ u &= 0 \quad \text{on } \partial\Omega, \end{aligned} \tag{4.2}$$

with the initial guess taken as a high frequency Fourier mode $u_0 = \sin(10\pi x)(10\pi y)$ is considered. The problem is tested using two different smoothers, the symmetric Gauss–Seidel and the Jacobi preconditioned Chebyshev smoothers. One application of the smoother is applied in terms of pre- and post- smoothing procedures of multigrid. The convergence obtained for both high and low quality meshes is shown in fig. 4.3. Clearly, for a high quality mesh, both methods work well, and the relative residual drops quickly. However, if the low quality mesh is applied, the convergence rate of using both smoothers decays significantly.

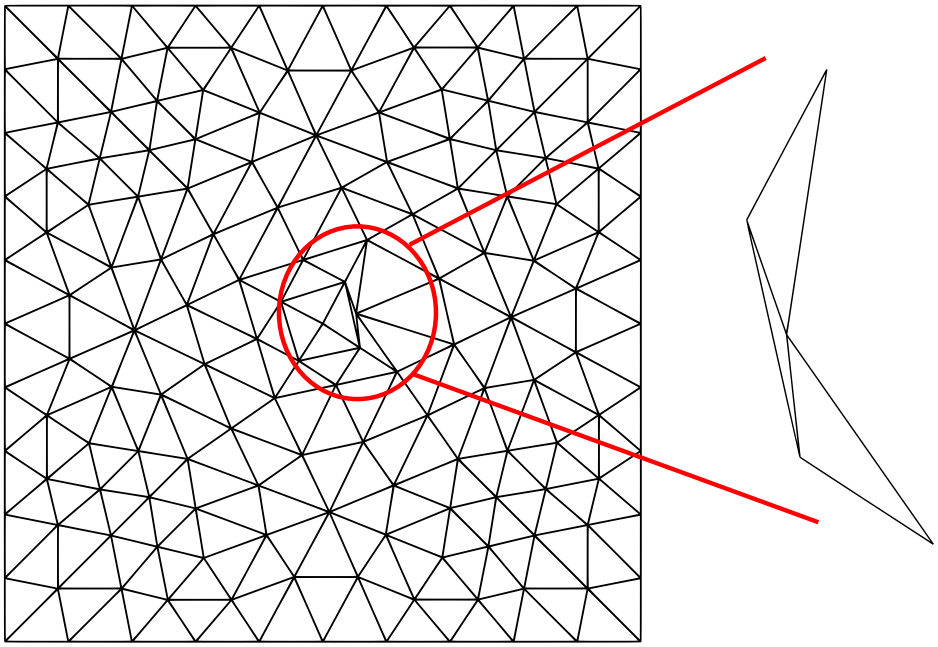
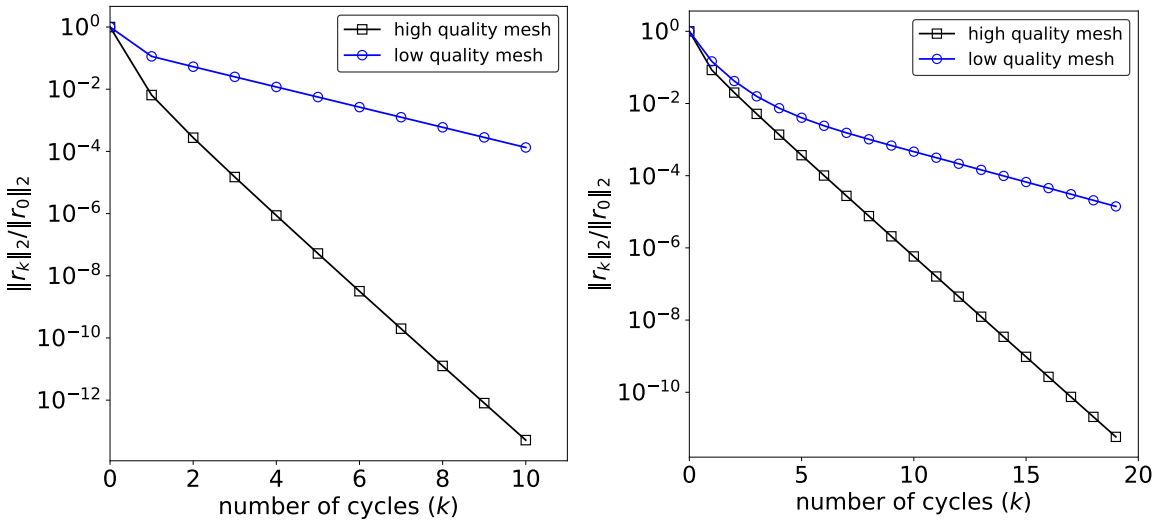


Fig. 4.2 A low quality unit square mesh with one poor quality region at the centre.



(a) Symmetric Gauss-Seidel

(b) Jacobi preconditioned Chebyshev

Fig. 4.3 Relative residual obtained by multigrid on high and low quality unit square meshes with (a) Symmetric Gauss–Seidel smoother and (b) Jacobi preconditioned Chebyshev smoother.

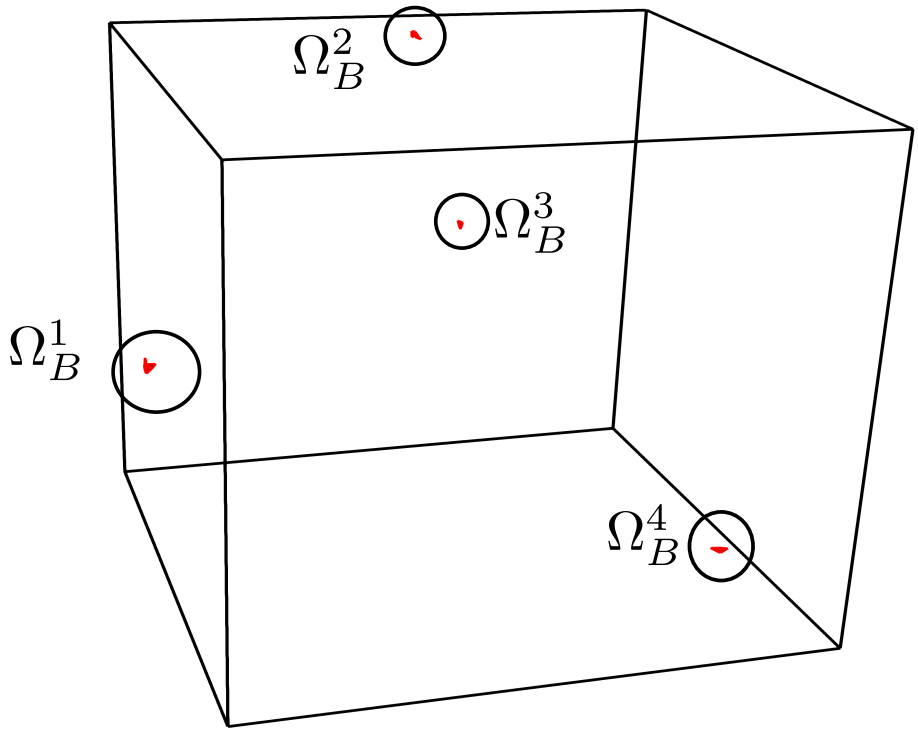


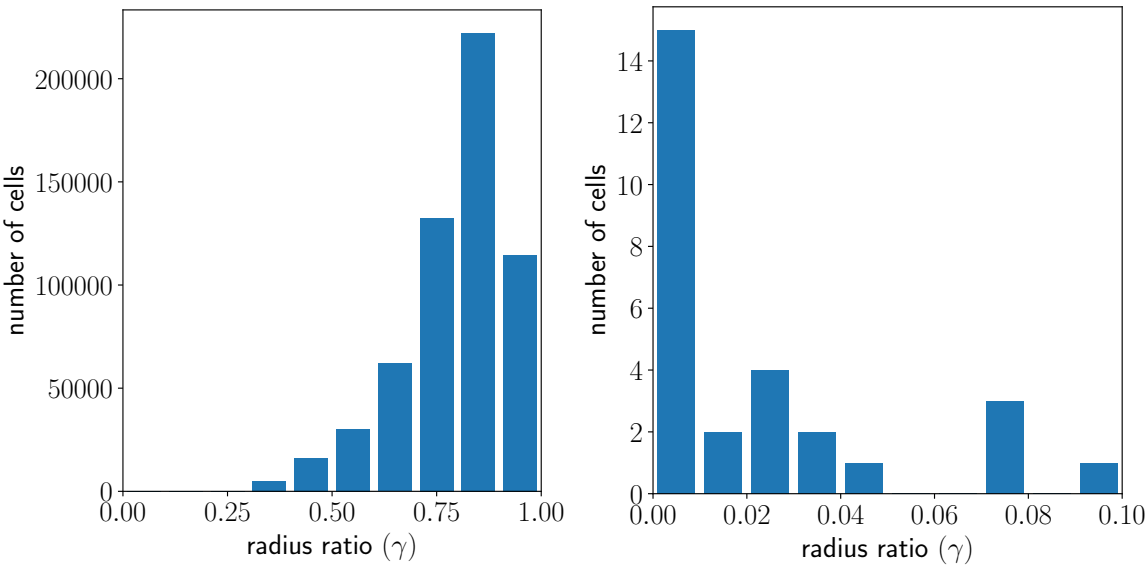
Fig. 4.4 Regions containing low quality cells on the finest low quality unit cube mesh.

In this example, the convergence is more influenced by low quality cells when using the Gauss–Seidel smoother since it has better smoothing property.

Example II Another example we test is a unit cube mesh with four regions of low quality cells. A four-level V-cycle is carried out for both P_1 and P_2 elements. The positions of the low quality cells on the finest grid (simulation level) is shown in fig. 4.4. In order to give a taste of the cell quality, a histogram of the normalised radius ratio (eq. (4.1)) of the finest grid is presented in fig. 4.5. It is clear that most parts of the domain are full of high quality cells (normalised radius ratio > 0.5). However, there are around 30 cells whose radius ratio is extremely small in the range of $0-0.1$. These elements whose radius ratio smaller than 0.1 can be enough to cause the unacceptable slow convergence of multigrid. A homogeneous Poisson equation with zero Dirichlet boundary condition is solved. The initial guess is taken by a high frequency mode,

$$u_0 = \sin(10\pi x)\sin(10\pi y)\sin(10\pi z). \tag{4.3}$$

Two cases are considered here, first the low quality cells appear on all grids, second the low quality cells only appear on the intermediate grids. The second case fits well the GMG configuration for only using the intermediate grids to faster the calculations. Two iterations



(a) Radius ratio of the unit cube

(b) Small radius ratio $(0-0.1)$ of the unit cube

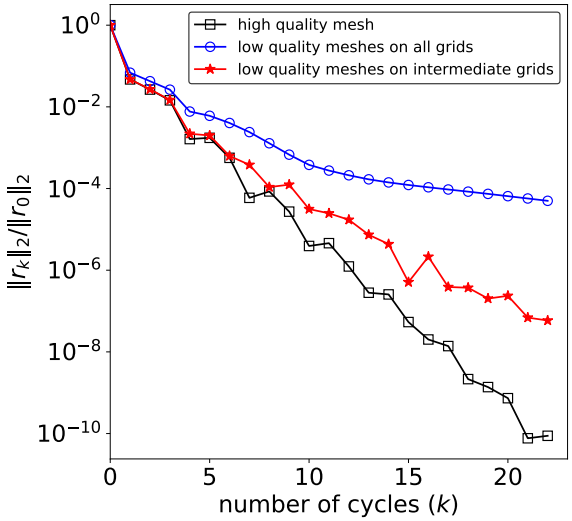
Fig. 4.5 Histogram of the normalised radius ratio of the finest low quality unit cube mesh.

of the symmetric Gauss–Seidel and Jacobi preconditioned Chebyshev methods are applied as smoothers for multigrid. The convergence results of relative residual are shown in fig. 4.6 for P_1 and P_2 elements.

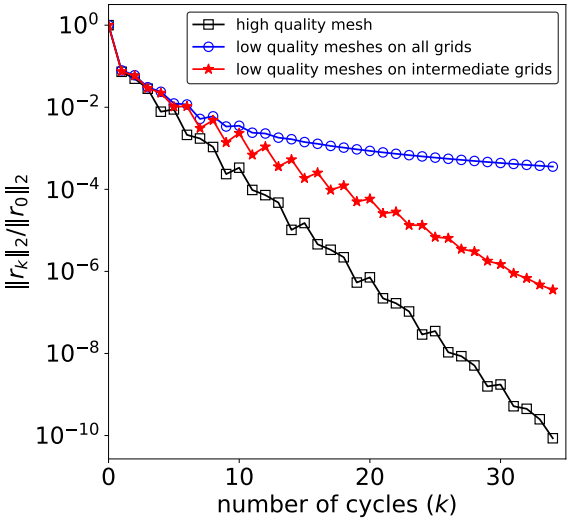
Apparently, the convergence is much slower if there exists any low quality mesh. And the low quality cells on the finest grid have a more detrimental influence on the convergence rate. For both smoothers, the influence of low quality cells on the convergence is stronger for using P_2 elements. Low quality cells only present on the intermediate grids do not totally destroy the convergence, but the convergence is still not acceptable compared to using the high quality mesh. The convergence rate obtained by using the Chebyshev smoother for P_2 element is the slowest.

4.2.2 Local failure of the smoothing property

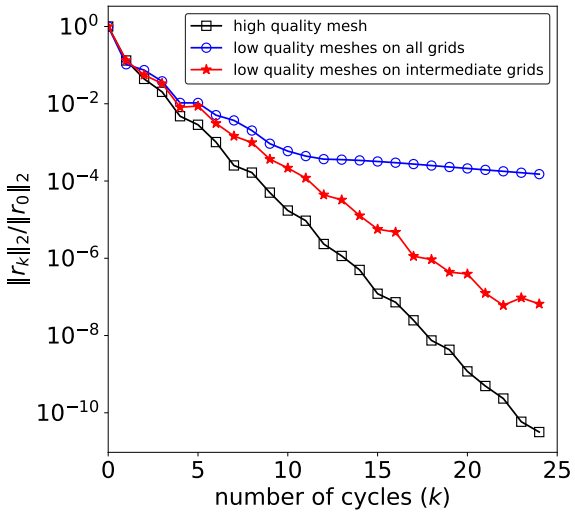
The reason behind the slow convergence of multigrid on low quality meshes is discussed here. The two examples in the previous section lead us to suspect the smoothing property of the smoothers in multigrid. We first go back to the unit square example shown in fig. 4.2. The same problem is solved with the high frequency initial guess, while here we only use the pure smoother to solve. Figure 4.7 gives the contour plots of the absolute value of error on each vertex after applying five and ten iterations of the symmetric Gauss–Seidel and Chebyshev smoothers. It is seen that after five iterations, the original high frequency error has been



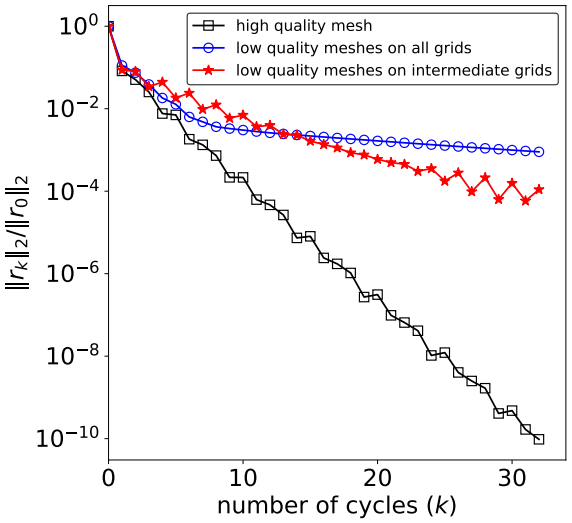
(a) Symmetric Gauss–Seidel for P_1 element



(b) Symmetric Gauss–Seidel for P_2 element

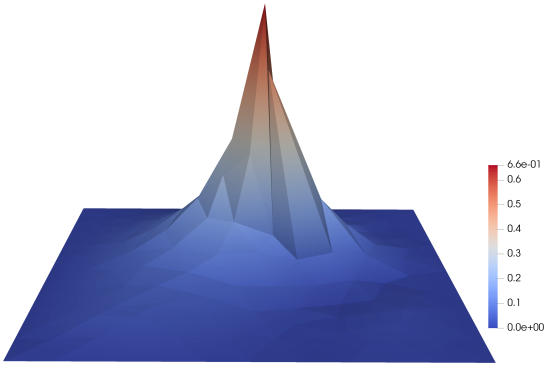


(c) Chebyshev for P_1 element

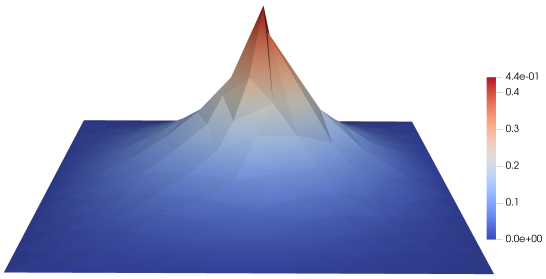


(d) Chebyshev for P_2 element

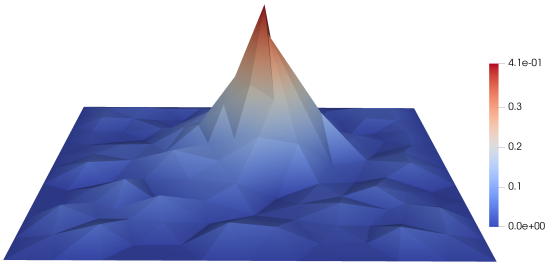
Fig. 4.6 Convergence rate of multigrid on the low quality unit cube mesh with Symmetric Gauss–Seidel and Jacobi preconditioned Chebyshev smoothers for P_1 and P_2 elements.



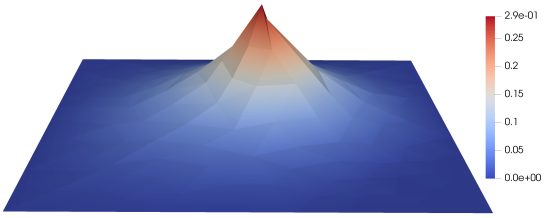
(a) Five iterations of Gauss–Seidel



(b) Ten iterations of Gauss–Seidel

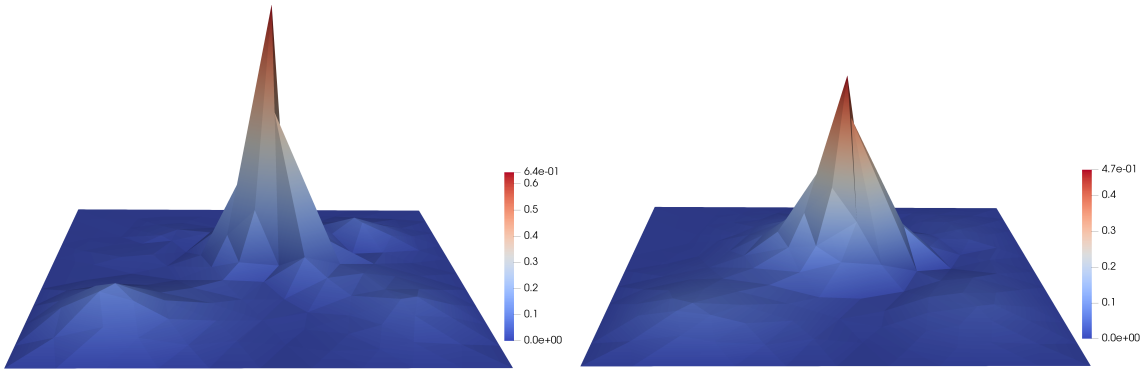


(c) Five iterations of Chebyshev



(d) Ten iterations of Chebyshev

Fig. 4.7 Absolute value of error on each vertex after five and ten iterations of symmetric Gauss–Seidel and Jacobi preconditioned Chebyshev methods on the low quality unit square mesh.



(a) Five iterations of conjugate gradient

(b) Ten iterations of conjugate gradient

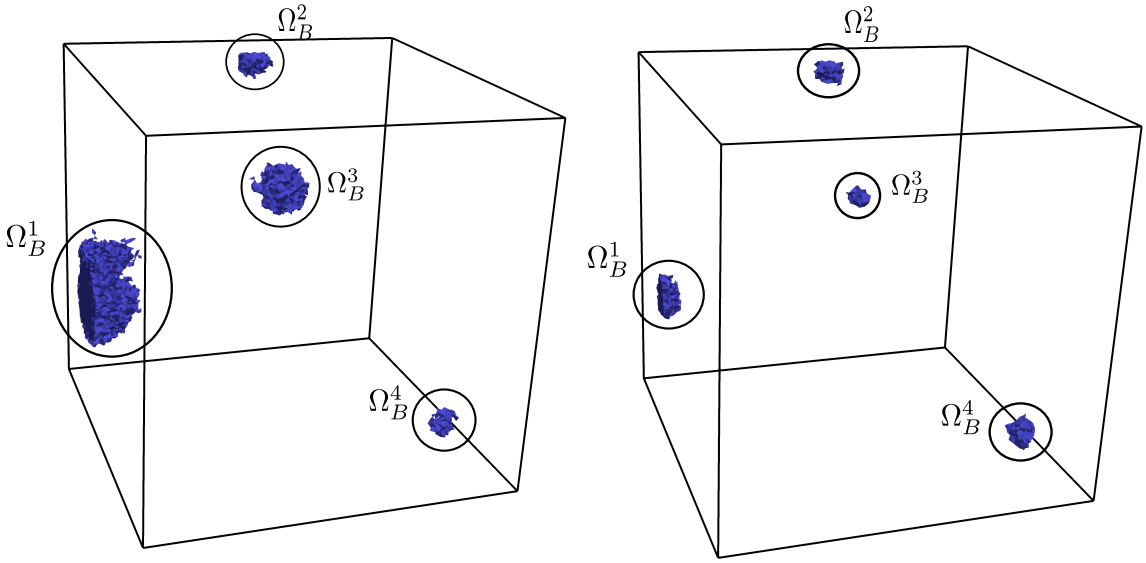
Fig. 4.8 Absolute value of error obtained by five and ten iterations of conjugate gradient method on the low quality unit square mesh.

successfully eliminated by both smoothers in most parts of region. However, there is some high frequency error appearing in the form of a localised peak, and this high frequency error is locally surrounded in the region of low quality cells. Moreover, this kind of high frequency error persists in the region of low quality cells. For both smoothers, this 'peak-like' error still remains without too much reduction even after ten iterations.

Actually it is observed that this phenomenon also happens if using other iterative methods or smoothers. The conjugate gradient method is applied to solve the same problem. Figure 4.8 shows the absolute value of error on each vertex after five and ten iterations of the conjugate gradient method on the low quality unit square mesh. The same high frequency error still appears and persists in the region of low quality cells. The defects in the mesh damages the smoothing property locally, namely the high frequency error can hardly be eliminated efficiently by the smoother.

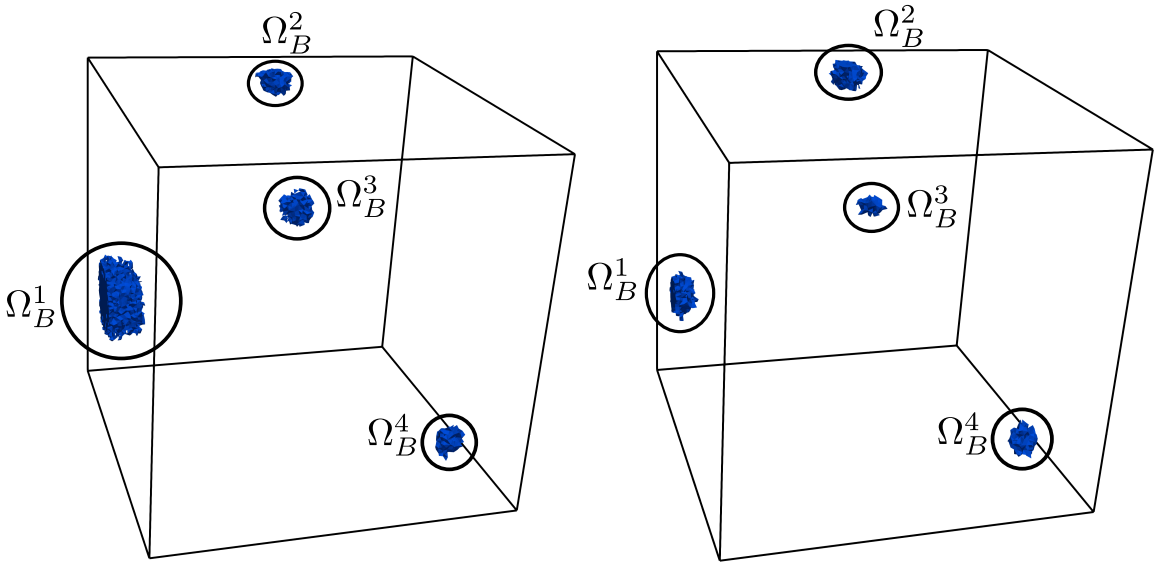
Now we consider the 3D problem in Example II, and again, the same homogeneous Poisson equation is solved with the high frequency initial guess. Pure symmetric Gauss–Seidel and Jacobi preconditioned Chebyshev methods are carried out. The regions containing large value of residual obtained by ten iterations of the methods for both P_1 and P_2 elements are shown in figs. 4.9 and 4.10. Though we can not see what exactly happens to the high frequency error, it is still obvious that those positions holding large values of residual collapse the regions of low quality cells. These large values of residual is hard to be removed by both iterative methods. There is no doubt that these persisting large residuals are the main reason that causes the slow convergence of multigrid.

The major conclusion to be reached here is that if a mesh with locally low quality cells is used, then the normal smoothing property fails in the region of low quality cells. The high



(a) P_1 element (b) P_2 element

Fig. 4.9 Locations holding the large value of residual on the low quality unit cube mesh after ten iterations of symmetric Gauss–Seidel method.



(a) P_1 element (b) P_2 element

Fig. 4.10 Locations holding the large value of residual on the low quality unit cube mesh after ten iterations of Jacobi preconditioned Chebyshev method.

frequency error persists in those regions, which causes the failure of the smoothers. We note that multigrid is nothing but nesting of these smoothers. The restriction of multigrid misses the high frequency error so that this kind of error can never be removed. This is the major reason that multigrid converges slowly or fails when using the locally poor quality meshes.

On the other hand, even though the smoother fails at the low quality area, it still works effectively on other parts of the domain. If those persisting high-frequency error in the low quality regions can be fixed, then the degraded performance of multigrid can be recovered. It leads to our idea of a local correction in low quality regions to enable multigrid to work with the meshes containing relatively small number of locally low quality cells (usually less than 10% of the total number of cells). A remark is made here that the local property of the smoother on the low quality mesh is not well studied here. Carrying out rigorous local Fourier analysis is the best way to understand the problem thoroughly, which is a topic of ongoing investigation.

4.3 A global–local smoother for multigrid

4.3.1 A Local correction smoother

Residual correction is still taken as the key to remove the high frequency error in low quality regions. The residual correction idea is usually viewed as a global procedure in many regimes including multigrid. For problems with locally poor quality meshes, there are only a few number of subdomains containing low quality cells, thus the residual correction can be employed as a local scheme. We aim to develop a local correction smoother to improve the local solution in regions of low quality cells. Suppose we are solving $Au = b$. Let u_k be the approximation obtained by k iterations of some iterative method, and r_k be the corresponding residual.

Suppose there are D subdomains containing locally poor quality cells. Denote them by Ω_B^d with $d = 1,2,\cdots,D$. Let B_d be the DOF set consisting all DOFs in the subdomain Ω_B^d with $B_d = \{\beta_i : i = 1,2,\cdots,n_B^d\}$, where $n_B^d := |B_d|$ is the number of DOFs in the subdomain. In practice, the set B_d is the same as the indices set of low quality region. It is required that these low quality subdomains are closed regions, and kept away from each other, namely

$$\begin{aligned} \bigcap_d \Omega_B^d &= \emptyset, \\ \bigcap_d B_d &= \emptyset. \end{aligned} \tag{4.4}$$

A local correction system on each subdomain Ω_B^d

$$A_c^d e_c^d = r_c^d, \tag{4.5}$$

is then generated, where A_c^d is the local submatrix of size $n_B^d \times n_B^d$, r_c^d is the corresponding local residual vector of size n_B^d, and e_c^d is the local error to be solved. The local submatrix A_c^d and the local residual r_c^d can be obtained by extracting the subsystem from the original system. The entries of A_c^d is obtained by taking the submatrix from A with respect to the DOF indices in B_d, namely

$$A_c^d[i,j] = A[\beta_i, \beta_j], \quad \forall \beta_i, \beta_j \in B_d, \quad i,j = 1,2,\cdots,n_B^d. \tag{4.6}$$

And the corresponding local residual r_c^d is obtained similarly,

$$r_c^d[i] = r_k[\beta_i], \quad \forall \beta_i \in B_d, \quad i = 1,2,\cdots,n_B^d. \tag{4.7}$$

To solve the local correction system, a direct solver, LU factorisation, is carried out. Since we assume the locally poor quality regions are small as well as the sets of indices B_d is small, then the direct solver solving the local correction system on all low quality regions can be cheap. Finally, we improve the solution by adding the corresponding local error, for $d = 1,2,\cdots,D$,

$$u_{k+1}[\beta_i] = u_k[\beta_i] + e_c^d[\beta_i], \quad \forall \beta_i \in B_d, \quad i = 1,2,\cdots,n_B^d. \tag{4.8}$$

The local correction smoother is explained in the algorithm 4.1.

Algorithm 4.1 Local correction smoother

1: **procedure** $u_{k+1} = S_c(A, b, u_k)$
2: Identify the poor quality regions Ω_B^d and their corresponding DOF set B_d, $d = 1,2,\cdots,D$
3: **for** $d = 1,2,\cdots,D$ **do**
4: Construct the local correction system A_c^d and r_c^d via eqs. (4.6) and (4.7).
5: Solve the local correction system $A_c^d e_c^d = r_c^d$ by the LU factorization.
6: Improve u_k by adding local errors via eq. (4.8).

Several remarks are drawn here. First, the local correction is easy to implement and its computational cost could be extremely cheap if the low quality regions are small as well as the corresponding sets of indices. Second, this step is more or less a domain decomposition method. It is anticipated that applying the direct solver on the low quality regions can

eliminate the high frequency remaining there. Finally, we can think of the local correction by solving a large matrix system containing the information from all low quality subdomains. Define the whole locally poor quality domain by

$$\Omega_B = \bigcup_d \Omega_B^d, \quad B = \bigcup_d B_d. \tag{4.9}$$

Rather than solving eq. (4.5) for all $d = 1, 2, \cdots, D$, a large system

$$A_c e_c = r_c, \tag{4.10}$$

can be solved where A_c is the block matrix of the size of $(\sum_{d=1}^{D} n_B^d) \times (\sum_{d=1}^{D} n_B^d)$ given by

$$A_c = A_c^1 \oplus A_c^2 \oplus \cdots \oplus A_c^D, \tag{4.11}$$

and the residual r_c has size of $(\sum_{d=1}^{D} n_B^d)$ given by

$$r_c = [r_c^1, r_c^2, \cdots r_c^D]^T. \tag{4.12}$$

And the correction step simply becomes

$$u_{k+1}[\beta_i] = u_k[\beta_i] + e_c[\beta_i], \quad \forall \beta_i \in B. \tag{4.13}$$

In this way, the local correction becomes a local smoother acting on the local domain Ω_B. It means that the local correction becomes a one-step smoother like the normal ones.

4.3.2 Global–local combined smoother

Local correction can certainly be used as a smoother for multigrid on local domains. For the problems with locally poor quality cells, we propose a new multigrid smoother which includes two steps. First, we apply a global smoother S_g, e.g. a symmetric Gauss–Seidel, on the whole domain; Then the local correction S_c smoother is applied on all the local subdomains Ω_B^d. Thus, our proposed smoother is a global–local combined smoother. The global smoother S_g on the whole domain eliminates the oscillatory error in regions of high quality cells $\Omega \setminus \Omega_B$ and ensures the effectiveness of multigrid. The local correction S_c removes those high frequency error which remains on the locally poor quality regions Ω_B and can not be eliminated by the global smoother S_g. In order to preserve the symmetric property of the multigrid, which can be essential for multigrid preconditioned conjugate

gradient method, the combined smoother is proposed in a 'sandwich form'. The multigrid with the proposed global–local combined smoother for solving the linear system eq. (3.18) is illustrated in algorithm 4.2.

Algorithm 4.2 Multigrid V-cycle with local correction

for $l = 1, 2, \cdots, L$ **do**
  procedure $u_l = Vcycle(A_l, b_l, u_l, l, \nu)$
    if $l \neq L$ **then**
      Pre-smoothing: apply ν times of combined smoother:
        (i) local residual correction $u_l \leftarrow S_c(A_l, b_l, u_l)$,
        (ii) global smoothing $u_l \leftarrow S_g(A_l, b_l, u_l)$,
        (iii) local residual correction $u_l \leftarrow S_c(A_l, b_l, u_l)$.
      Calculating the residual $r_l = b_l - A_l u_l$.
      Coarse grid construction $A_{l+1} = P_l^T A_l P_l$, $b_{l+1} = P_l^T r_l$ and $u_{l+1} = P_l^T u_l$.
      Coarse grid correction $u_{l+1} = Vcycle(A_{l+1}, b_{l+1}, u_{l+1}, l+1, \nu)$.
      Updating current solution $u_l \leftarrow u_l + P_l u_{l+1}$.
      Post-smoothing: apply ν times of combined smoother:
        (i) local residual correction $u_l \leftarrow S_c(A_l, b_l, u_l)$,
        (ii) global smoothing $u_l \leftarrow S_g(A_l, b_l, u_l)$,
        (iii) local residual correction $u_l \leftarrow S_c(A_l, b_l, u_l)$.
    else
      Direct solver on the coarsest grid $u_L = A_L^{-1} b_L$.

This kind of global–local idea has been applied to solve anisotropic elliptic PDEs, see [21, 38]. It is shown in the next section that this form of the combined smoother is symmetric. This multigrid with the new smoother is still a linear symmetric iterative solver if the global smoother is linear. Therefore, we can still employ multigrid with local correction as a preconditioner for standard conjugate gradient method.

4.3.3 Identifying local correction regions

The mesh quality measure plays a crucial role in finding the areas of low quality cells. As discussed, there is no direct clue relating the mesh quality and convergence of an iterative method. A straightforward way to track low quality regions Ω_B is to set a threshold on some mesh quality. Here, we first track the literally low quality cells by finding all elements τ whose radius ratio is smaller than a certain threshold say ε, namely, find

$$\Omega_b = \{\tau : \ \gamma(\tau) < \varepsilon\},$$

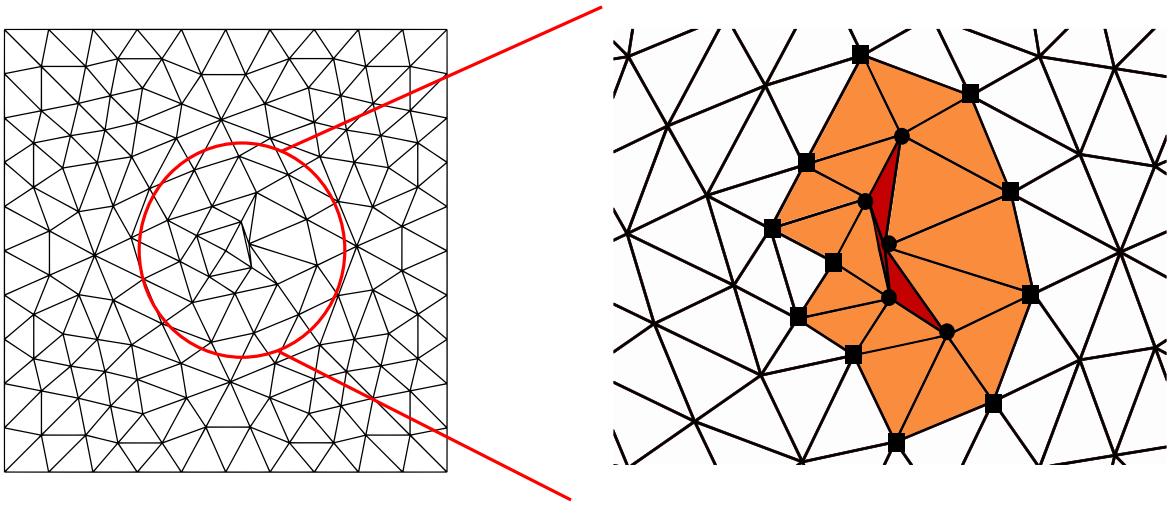


Fig. 4.11 Identify the local correction region Ω_B for the low quality unit square mesh in fig. 4.2, the area coloured red contains the low quality cells in Ω_b, and the whole coloured area becomes the local correction region Ω_B.

where γ is the normalised radius ratio in eq. (4.1) and ε is set to be 0.1 in this chapter. It is observed that the high frequency error generated by the region Ω_b may diffuse out, thus Ω_b may not be big enough to cover entire area containing all high frequency error. The local correction region Ω_B, in which we apply the local correction smoother, is then chosen by extending one layer of cells in Ω_b, i.e.,

$$\Omega_B = \Omega_b \cup \{\tau : \tau \text{ shares edges/nodes with } \Omega_b\}.$$

Note that local correction is sensitive with respect to the choice of the region Ω_B. A bad choice of Ω_B can slow down the convergence dramatically. It is difficult to predict how large the subdomain Ω_B should be to guarantee a satisfied convergence rate of multigrid. Extending one layer of bad cells in low quality region Ω_b is found to be well adequate in our numerical examples.

We use the unit square mesh shown in fig. 4.2 to make an illustration. The local correction region is identified in fig. 4.11. The low quality cells in Ω_b with radius ratio $\gamma < 0.1$ are coloured red, which are the three triangles at the centre. Then those cells coloured brown are the one layer extension of cells in Ω_b. Thus, the local correction region Ω_B is chosen by the whole coloured area.

It is noted that a big challenge of this method is the high computational cost of local correction when the area of low quality subdomain Ω_B is large. If there are a large number of low quality cells, then the cost of local correction would be even higher than the global smoother. At this time, local correction may not be applicable. However, we can decrease

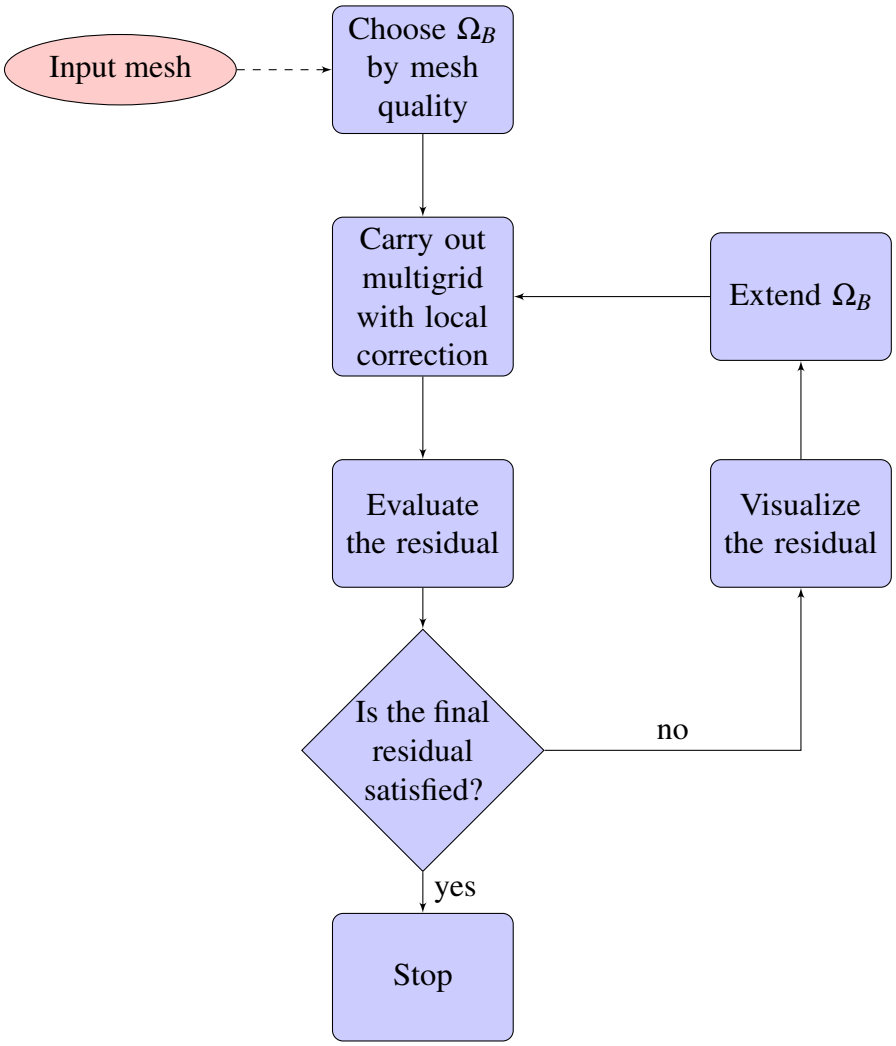


Fig. 4.12 Work flow of multigrid on low quality meshes.

the number of low quality cells by tracking those cells that have the most significant impact on the convergence. A specific example is shown in chapter 8, in which we find that those cells close to the turbines are the main cause for the slow convergence of multigrid.

In practice, Ω_B can be hard to be chosen *a priori*. It is difficult to predict if the size of Ω_B is large enough to ensure the convergence at the beginning. On the other hand, it is suggested that the local correction region can be chosen *a posteriori*. The local correction region can be chosen by the value of residual observed. We have seen, in the previous section, that the areas with large value of residual collapse the regions of low quality cells. Therefore, the positions where large value of residual lies should be included in Ω_B. A good work flow of this idea is shown in fig. 4.12.

We leave a remark here that it is possible to automate the proposed work flow if we set certain thresholds. An original threshold can be set to determine whether or not the calculation result is satisfied, for example, we want relative residual less than 10^{-10}. If the result is not satisfied, we can track cells with large components of residual by setting another threshold, for instance, every time those cells with the largest 10% residual can be chosen to go through the local correction. By doing so, the whole process become automated.

4.3.4 Corrected largest eigenvalue for Chebyshev smoother

We make a remark here for using the Chebyshev smoother for its special smoothing property. The smoother formula for the Chebyshev method, described in section 2.5, can be written as

$$S = \sum_{j=0}^{n} \alpha_j A^j, \tag{4.14}$$

where α_j are coefficients derived from the Chebyshev polynomial. The terms α_j depend on the largest and the smallest eigenvalues of the system to which it is applied. The use of Chebyshev iteration as a smoother requires estimation of the largest eigenvalue [1]. However, a region of low quality cells can dramatically increase the maximum eigenvalue relative to a high quality mesh, and destroy the smoothing properties of the Chebyshev iteration on a grid. With the 'wrong' largest eigenvalue input, the smoothing property of the Chebyshev method becomes ineffective on the high quality regions, which brings a global problem. Thus, we choose to use a corrected largest eigenvalue input based on the high quality area $\Omega \setminus \Omega_B$. Suppose the system is decomposed into two parts with

$$A = \begin{bmatrix} A_{gg} & A_{gb} \\ A_{bg} & A_{bb} \end{bmatrix},$$

where A_{gg} and A_{bb} correspond to the high quality and low quality parts of the matrix respectively. The corrected largest eigenvalue with respect to the high quality region is taken as $\lambda_{\max} \leftarrow \lambda_{\max}(A_{gg})$. This approximation is valid if the low quality region is much smaller than the high quality region. The strategy is tested numerically in section 4.5.

4.4 Relation to the domain decomposition method

We now present an alternative way to view our global–local combined smoother. It is found that the scheme is equivalent to certain domain decomposition method and in the form of the block Gauss–Seidel. Analytical formulations of the proposed smoother are also given

in this section. Suppose we are solving $Au = b$, where $A \in \mathbb{R}^{n\times n}$. Let u_k be some initial approximation to the linear system, and the corresponding error and residual be e_k and r_k, respectively.

4.4.1 Local correction

We formulate the local correction smoother S_c here. Let I_d be the natural inclusion mapping from the local subdomain Ω_B^d to the whole domain Ω, which is a $n \times n_B^d$ matrix given by

$$I_d[i,j] = \begin{cases} 1 & i = \beta_j, \quad \beta_j \in B_d, \\ 0 & \text{Otherwise.} \end{cases} \tag{4.15}$$

In practice, the function I_d maps set of indices in Ω_B^d to set of indices in Ω. The local error and residual can be thought of as the restriction of the global value to the local subdomain,

$$e_c^d = e_k|_{\Omega_B^d} = I_d^T e_k,$$
$$r_c^d = r_k|_{\Omega_B^d} = I_d^T r_k.$$

Then the local correction system can be generated by

$$A_c^d = I_d^T A I_d.$$

This formulation gives the submatrix extracted from the original matrix with respect to DOFs in the subdomain, i.e. eq. (4.6). The following lemma shows the nonsingularity of the local correction system.

Lemma 4.1. *If A is symmetric positive definite (SPD), then A_c^d is SPD for all $d = 1, 2, \cdots, D$*

Proof. The symmetry follows directly from the symmetry of A. For all vectors $v \in \mathbb{R}^{n_B^d} \setminus \{0\}$,

$$v^T A_c^d v = v^T (I_d)^T A I_d v = (I_d v) \cdot (A I_d v) = (A I_d v) \cdot (I_d v) \geq 0,$$

since A is SPD. On the other hand, if we have $v^T A_c^d v = 0$, then $I_d v = 0$. Decompose the vector v in space $\mathbb{R}^{n_B^d}$ by orthogonal unit vectors e_i so that $v = \sum_{i=1}^{n_B^d} e_i$. Then

$$I_d \left(\sum_{i=1}^{n_B^d} e_i \right) = \sum_{i=1}^{n_B^d} I_d e_i = 0,$$

holds if and only if $e_i = 0$ for all i since I_d is a non-zero matrix. □

In fact, the numerical examples in section 4.5 also validate that the local correction matrix is nonsingular with respect to different boundary conditions and element types. The local correction solves the residual equation on the subdomain Ω_B^d directly,

$$e_c^d = (I_d^T A I_d)^{-1} I_d^T r_k,$$

and via the inclusion, the local error can be mapped back to the global domain. The solution u_k is then corrected by adding the error parts on all the subdomains,

$$u_{k+1} = u_k + \sum_{d=1}^{D} I_d e_c^d.$$

The local correction smoother can be written in a preconditioner form with

$$u_{k+1} = u_k + S_c \left(b - A u_k\right),$$

where

$$S_c = \sum_{d=1}^{D} I_d (I_d^T A I_d)^{-1} I_d^T. \tag{4.16}$$

This formula is familiar to us, it is equivalent to the subspace correction methods described in section 2.6.3. In particular, the low quality region can be viewed as a (local) subspace of the whole domain. The local correction S_c just corrects the error on the local subspace. Algebraically, the local correction smoother is equivalent to the parallel subspace correction applied on the local subdomain with low quality cells. In terms of implementation, it is easy to implement local correction in parallel by solving each subsystem on one processor.

4.4.2 Global–local combined smoother

As discussed above, the multigrid smoother we propose is nothing but a combination of a global smoother S_g on the whole domain Ω and the local correction smoother S_c on the local subdomain Ω_B. And the combined smoother is in a sandwich form which perseveres the symmetry. If we split the three step of the combined smoother, then it can be written as

$$\begin{aligned} u_{k+1/3} &= u_k + S_c \left(b - A u_k\right), \\ u_{k+2/3} &= u_{k+1/3} + S_g \left(b - A u_{k+1/3}\right), \\ u_{k+1} &= u_{k+2/3} + S_c \left(b - A u_{k+2/3}\right). \end{aligned} \tag{4.17}$$

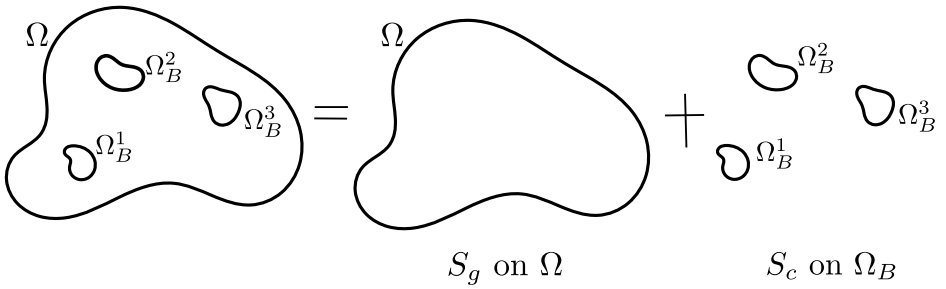


Fig. 4.13 Domain decomposition view of the combined global-local smoother.

Rearrange the formula, the preconditioner form of the combined smoother becomes

$$u_{k+1} = u_k + S_{gc}\,(b - Au_k)\,, \tag{4.18}$$

where

$$S_{gc} = S_c + (I - S_c A) S_g (I - A S_c). \tag{4.19}$$

It can be seen that the smoother we propose is symmetric, which is well-suited for multigrid preconditioned conjugate gradient.

This procedure reminds us of the Schwarz type domain decomposition method described in section 2.6.1. The global–local combined smoother can be viewed as a domain decomposition method by cutting the domain into two parts: the whole domain and several small subdomains. Suppose we think of the whole domain as the first 'subdomain', then the domain decomposition is based on

$$\Omega = \Omega \cup \{\Omega_B^1 \cup \cdots \cup \Omega_B^D\} = \Omega \cup \Omega_B. \tag{4.20}$$

Therefore, this combined smoother can be thought of as a full overlapped Schwarz type domain decomposition method [103]. Figure 4.13 gives an illustration of the combined smoother. On the other hand, from the algebraic view, the combined smoother is also in the form of a block Gauss–Seidel method, formulated in section 2.6.2. The system, the combined smoother solves, is then a partition of the whole domain and local subdomains,

$$\begin{bmatrix} A & & & \\ & A_c^1 & & \\ & & \ddots & \\ & & & A_c^D \end{bmatrix} = \begin{bmatrix} A & \\ & A_c \end{bmatrix}.$$

It can be thought of that we apply the block Gauss–Seidel to solve the above system via using a global smoother to solve matrix A and a direct solver to solve the local system A_c. This type of block Gauss–Seidel can be implemented efficiently in parallel [6, 101, 3]. A proposed way of parallel implementation of the global–local smoother is given in section 9.2.3.

The proposed combined smoother is similar to certain plane smoothers [108, 78, 77], which are variants of block Gauss–Seidel. The plane smoother applies standard smoother on multi-block structured grids like x- or y-planes. The plane smoother is well-known for its high efficiency for strongly anisotropic problems, especially for fluid problems, e.g. [88]. Our combined multigrid smoother solves the problem in the order of first the whole domain, then the local subdomains. Therefore, it can be considered as a (global,local)-plane Gauss–Seidel with direct solver in each local plane.

4.5 Numerical examples

We examine numerically the performance of the multigrid with the proposed smoother for Poisson and elasticity problems using the finite element method. Lagrange elements are used, with linear (P_1) and quadratic (P_2) bases on simplices. Pre- and post-smoothing are used with $\nu = 2$ iterations for all examples except the first simple unit square mesh one. A direct solver is used on the coarsest grid. The solver is terminated once the relative residual reaches 10^{-10} measured in the 2-norm.

All meshes (grids) are unstructured with simplex cells, and the levels are non-nested. We restrict ourselves to problems in which the geometry can be exactly represented by the coarsest grid. Meshes for each level are generated using Gmsh [48]. To examine the influence of mesh quality and to model the effect of low quality cells in complex problem, we move some vertices in the generated meshes to modify, controllably, the mesh quality characteristics. We identify low quality regions as regions where $\gamma \leq 0.1$, where γ is the normalised radius ratio in eq. (4.1). Denote γ_{min} as the minimum radius ratio, Ω_{high} and Ω_{low} for the high and low quality meshes respectively. We consider cases where all grids (meshes) have low quality regions (case A), and cases where the fine grid is of high quality and the intermediate levels have low quality regions (case B). In particular, the performance of solvers on high quality meshes for all examples is also shown as a reference case to compare.

The examples are implemented using libraries from the FEniCS Project [5, 80, 79] and PETSc [11, 10, 12]. The Galerkin finite element systems and the restriction/prolongation operators are constructed with FEniCS, and the remaining generic multigrid functionality

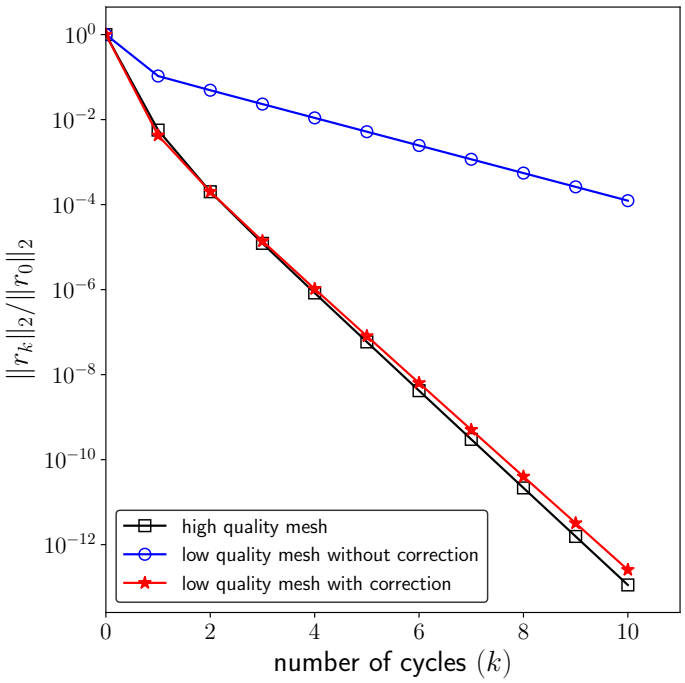
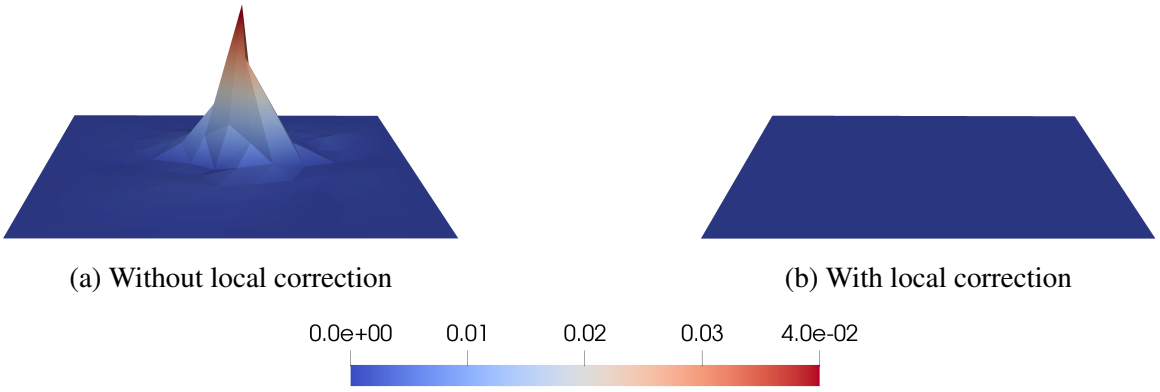


Fig. 4.14 Relative residual obtained by multigrid with and without the local correction for the Poisson problem on a unit square.

is provided by PETSc. The source code of implementation is freely available as part of the supporting material [29].

4.5.1 Poisson problem on a unit square

The homogeneous Poisson problem in eq. (4.2) on the unit square is investigated again, with the initial guess of $u^{(0)} = \sin(10\pi x)\sin(10\pi y)$ and solved by a two-level V-cycle. The fine grid after perturbation of some vertices is shown in fig. 4.2, and has 272 cells. The coarse grid has 68 cells, and remains high quality. One application of the symmetric Gauss–Seidel smoother is used in the pre- and post-smoothing steps. The residual after each multigrid cycle is shown in fig. 4.14 for (i) the poor quality fine grid with the standard smoother, (ii) the poor quality fine grid with the local correction, and (iii) a high quality fine grid (reference solution). The poor convergence with the low quality mesh is clear, and the local correction recovers the convergence rate of the reference solution. Figure 4.15 shows the absolute value of the solution error on the poor quality fine grid after five V-cycles with and without the local correction. It is clear that the error persists in a localised region around the low quality cells without the local correction, and this error is removed by the local correction smoother.



(a) Without local correction (b) With local correction

Fig. 4.15 Absolute value of the error after five multigrid cycles on the low quality unit square mesh with and without local correction.

level	$\gamma_{\min}(\Omega_{\text{high}})$	$\gamma_{\min}(\Omega_{\text{low}})$	number of cells in Ω	number of cells in Ω_B	number of DOFs in Ω P_1/P_2	number of DOFs in Ω_B P_1/P_2
1 (fine)	0.661	1.53×10^{-4}	4236	49	2199/8633	44/134
2	0.673	3.10×10^{-4}	1016	41	549/2133	37/112
3	0.772	2.09×10^{-4}	254	36	148/549	34/100
4 (coarse)	0.773	9.08×10^{-4}	68	11	45/157	10/30

Table 4.1 Mesh quality and mesh sizes of the unit square hierarchy meshes as well as the sizes of the local correction systems.

We consider now a four-level V-cycle, with symmetric Gauss–Seidel used as the global smoother. The finest grid is shown in fig. 4.16 with the positions of low quality cells. The number of cells for each level, and the number of degrees-of-freedoms for P_1 and P_2 elements, are presented in table 4.1. Table 4.1 also summarises the quality and sizes of the degraded mesh regions. The minimum radius ratio is of the order 10^{-4}. The fraction of low quality cells is small. The tests are for the problem

$$\begin{aligned} -\nabla^2 u &= 2\pi^2 \cos(\pi x)\sin(\pi y) \quad \text{in} \quad \Omega = (0,1)^2 \\ u &= 0 \quad \text{on} \quad \Gamma_D = \{(x,y) \in \partial\Omega : y = 0,1\} \\ \nabla u \cdot n &= 0 \quad \text{on} \quad \Gamma_N = \{(x,y) \in \partial\Omega : x = 0,1\}. \end{aligned} \tag{4.21}$$

The reduction in the residual is considered for the case of poor quality cells on all levels (case A) and poor quality cells on levels other than the finest level (case B). Figure 4.17 shows the computed relative residual for these two scenarios with and without local correction for P_1 and P_2 elements. It is clear that the convergence rate is slow for low quality meshes, and

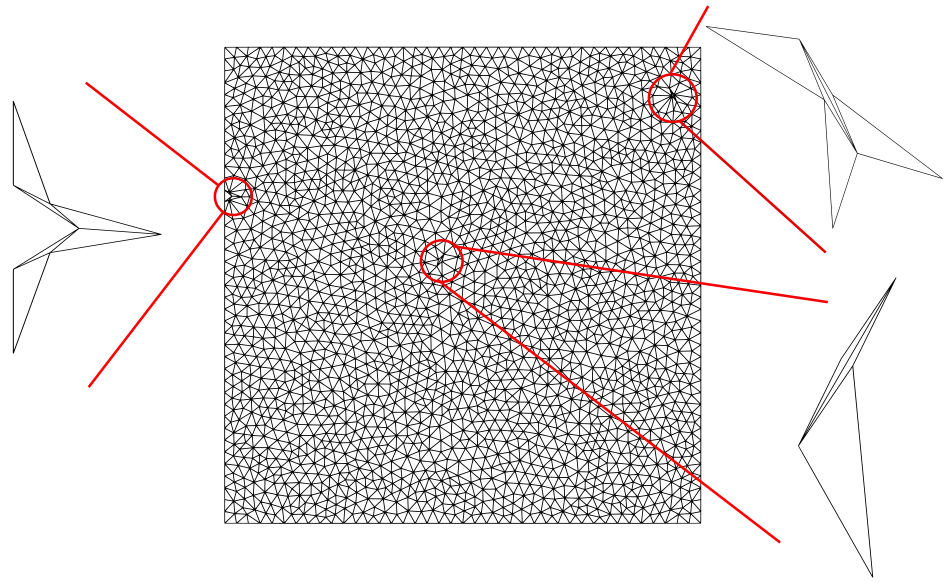


Fig. 4.16 A unit square mesh with three regions of low cell quality.

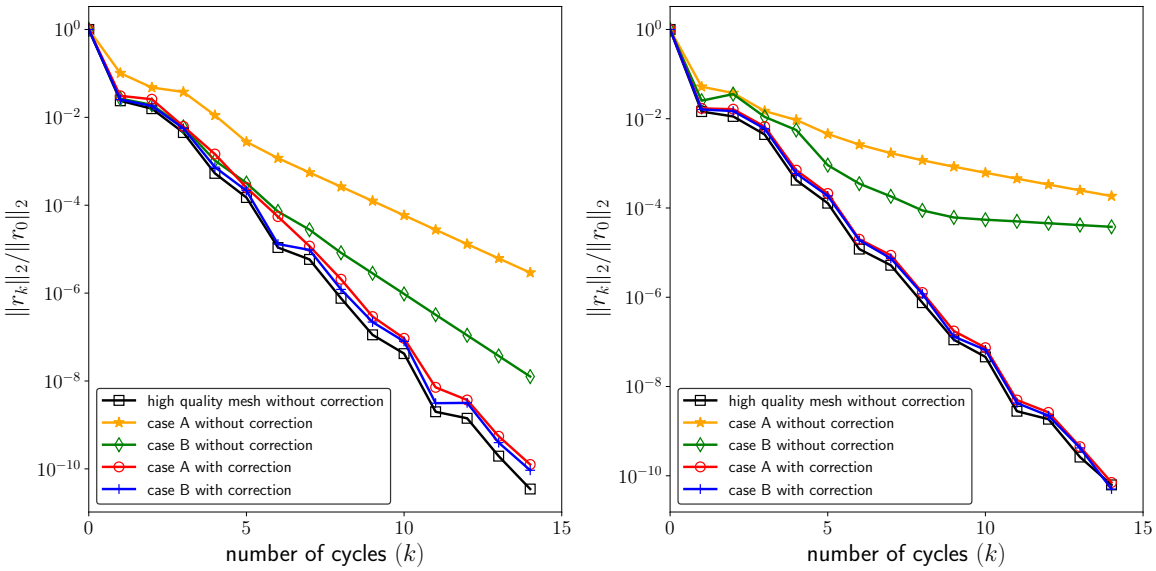
solver error	$\|u_{12} - u^*\|_2$			
element type	P_1		P_2	
case	case A	case B	case A	case B
high quality mesh	2.86×10^{-8}	2.86×10^{-8}	7.04×10^{-9}	7.04×10^{-9}
low quality mesh without correction	2.09×10^{-6}	1.57×10^{-7}	3.91×10^{-5}	3.76×10^{-5}
low quality mesh with correction	8.04×10^{-8}	6.35×10^{-8}	1.10×10^{-8}	1.01×10^{-8}

Table 4.2 Error in the 2-norm after 12 multigrid cycles with and without local correction, where u_{12} is the multigrid solution after 12 cycles and u^* is the exact solution to the linear system obtained by a direct method.

particularly so for quadratic elements. In both cases the local correction smoother recovers the convergence rate of the reference solution with high quality meshes.

Table 4.2 lists the 2-norm of the exact solver error after 12 cycles of multigrid with and without local correction. The error obtained without local correction can be large for both elements and both cases, which is due to the high frequency error remaining in low quality cells. The local correction smoother can not only reduce the residual but also reduce the error of the solver.

On the other hand, the finite element solution error in the L_2 norm is investigated. Table 4.3 lists the finite element error for both cases and both element types. Without local correction, there is an increase in the error if existing any low quality meshes, especially for the case A. With local correction, the finite element error is reduced. In particular, for the



(a) P_1 elements (b) P_2 elements

Fig. 4.17 Relative residual for the Poisson problem on the unit square domain with low quality regions on all levels (case A) and with low quality regions on all levels except the finest level (case B).

case B in which we only encounter low quality intermediate grids, the accuracy of the FEM solution is equivalent to the high quality mesh case.

In fig. 4.18, we show the absolute value of residual on the finest grid after 10 V-cycles for the case of low quality cells on all levels and with the local correction on coarse grids only. Unsurprisingly the residual persists in the regions with low quality cells.

4.5.2 Poisson problem on a unit cube

We consider now the three-dimensional test problem on the unit cube $\Omega = (0,1)^3$:

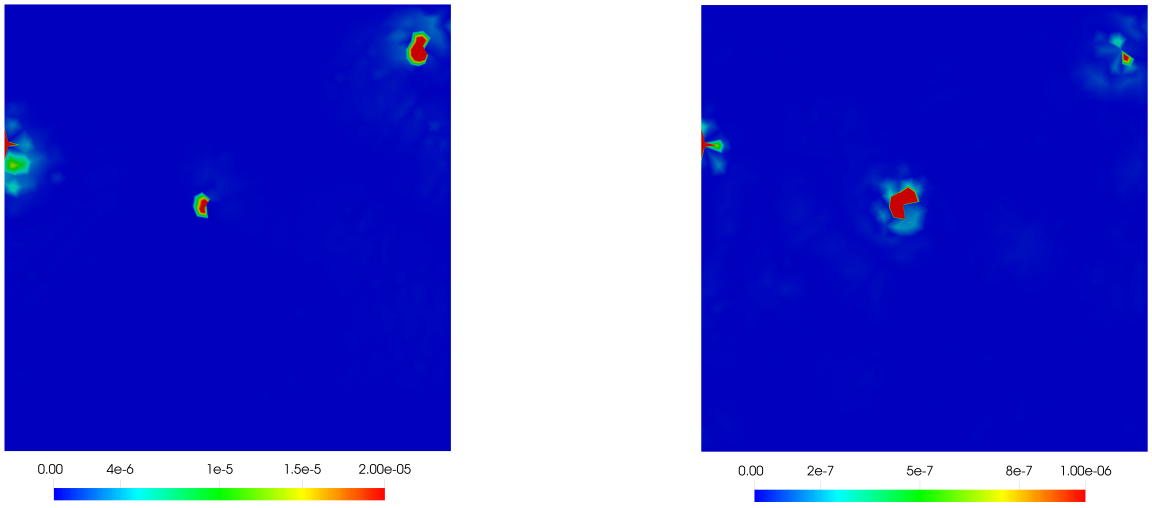
$$\begin{aligned} -\nabla^2 u &= 10\exp\left[-\left((x-0.5)^2+(y-0.5)^2+(z-0.5)^2\right)/0.02\right] \quad \text{in } \Omega, \\ u &= 0 \quad \text{on } \Gamma_D = \{(x,y,z) \in \partial\Omega : x = 0,1\}, \\ \nabla u \cdot n &= \sin(5x)\sin(5y) \quad \text{on } \Gamma_N = \partial\Omega \setminus \Gamma_D. \end{aligned}$$

We use the four-level multigrid V-cycle, which is the same in Example II shown in section 4.2.1. The coarsening rate in terms of number of degrees-of-freedom is chosen in the range of 7–8. The mesh quality and mesh sizes of levels are summarised in table 4.4.

The reduction in the relative residual with each multigrid cycles using a symmetric Gauss–Seidel smoother, with and without the local correction, is shown in fig. 4.19. Again the

FEM error	$\|u_{12} - u^*\|_{L_2(\Omega)}$			
Element type	P_1		P_2	
Case	case A	case B	case A	case B
High quality mesh	4.35×10^{-4}	4.35×10^{-4}	2.56×10^{-6}	2.56×10^{-6}
Low quality mesh without correction	2.09×10^{-3}	5.25×10^{-4}	3.91×10^{-5}	4.39×10^{-6}
Low quality mesh with correction	7.55×10^{-4}	4.35×10^{-4}	6.61×10^{-6}	2.56×10^{-6}

Table 4.3 Finite element Error in the L_2-norm after 12 multigrid cycles with and without local correction, where u_{12} is the approximated solution and u^* is the exact solution to the PDE.

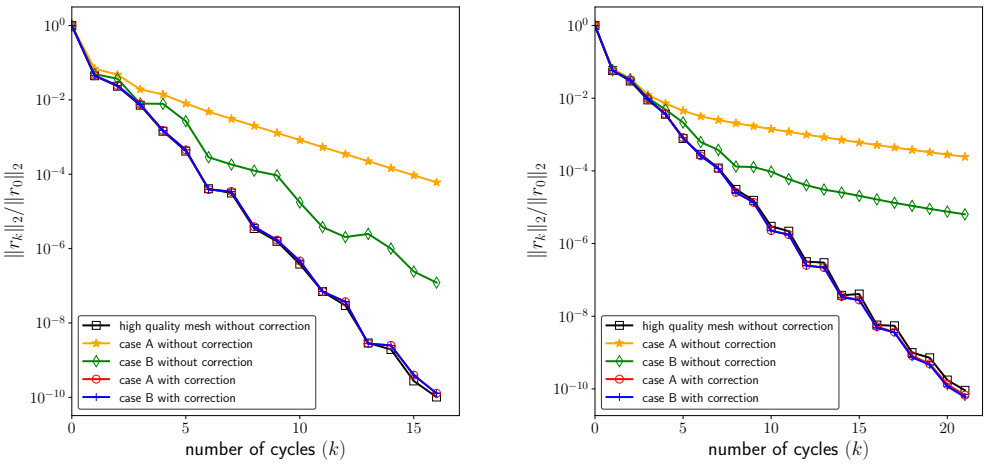


(a) P_1 element (b) P_2 element

Fig. 4.18 Absolute value of residual on each vertex of the finest grid of the unit square after ten cycles of multigrid without local correction for P_1 and P_2 elements.

level	$\gamma_{\min}(\Omega_{\text{high}})$	$\gamma_{\min}(\Omega_{\text{low}})$	number of cells in Ω	number of cells in Ω_B	number of DOFs in Ω P_1/P_2	number of DOFs in Ω_B P_1/P_2
1 (fine)	0.275	7.20×10^{-6}	582730	625	104976/814775	231/1262
2	0.288	2.77×10^{-6}	65259	604	13361/97422	244/1294
3	0.278	1.74×10^{-6}	7165	568	1776/11845	233/1226
4 (coarse)	0.336	4.64×10^{-6}	792	116	251/1501	50/258

Table 4.4 Mesh level summary for the unit cube domain.



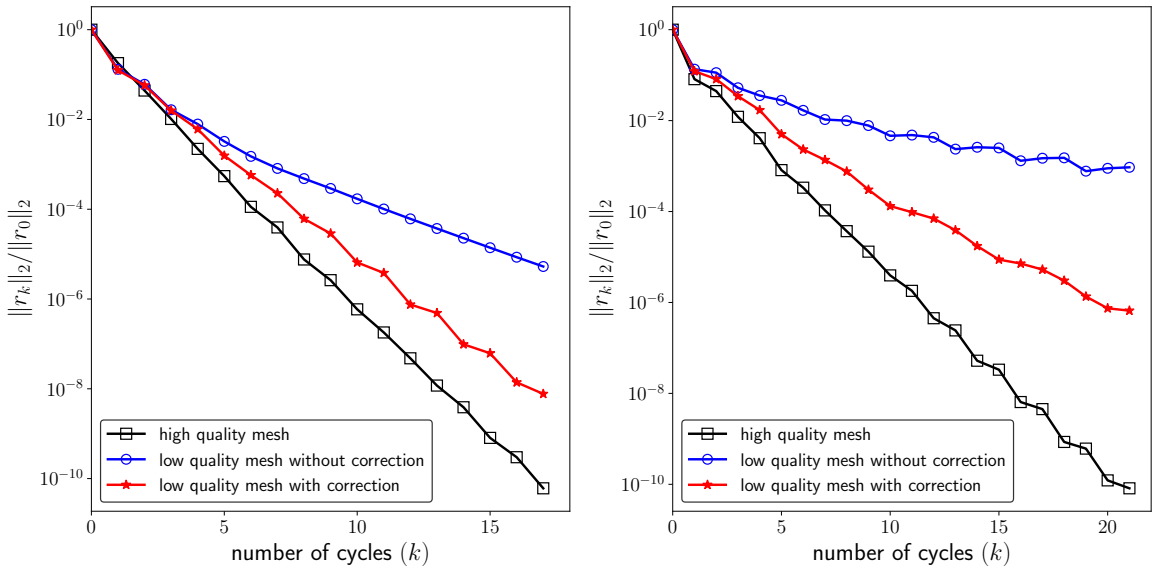
(a) P_1 element (b) P_2 element

Fig. 4.19 Relative residual obtained by multigrid with symmetric Gauss–Seidel smoother carried out on the unit cube meshes with low quality regions on all levels (case A) and with low quality regions on all levels except the finest level (case B).

convergence rate slows for cases with a low quality level, which is particularly pronounced with P_2 elements. And the local correction restores the convergence rate to the high quality mesh reference case.

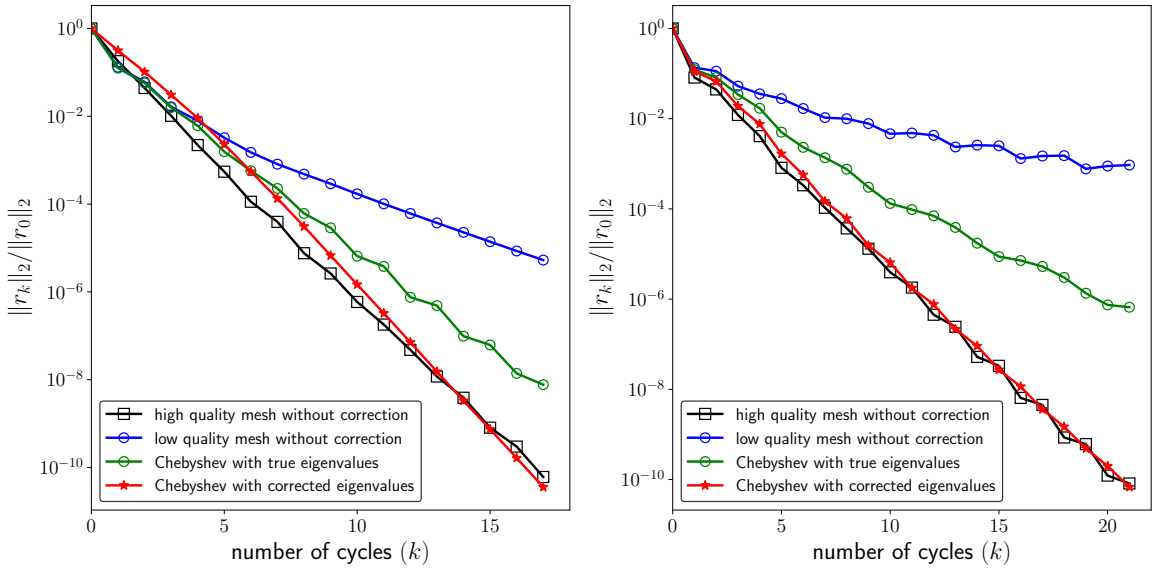
We consider now the Chebyshev smoother place of symmetric Gauss–Seidel to test the problem for case A in which all levels contain low quality cells. Recall that the Chebyshev method requires an estimate of the extreme eigenvalues. We first use the unshifted largest eigenvalue of the whole system to construct the smoother. The relative residual results with cycle counts are shown in fig. 4.20. It is found that the poor performance also happens for Chebyshev smoother, and the local correction does help improve the convergence rate of the low quality meshes. However, there is still some gap to get a full recovery like in the Gauss–Seidel case, namely, the convergence rate at reference level is still not achieved. The reason is the the smoothness effects of Chebyshev smoother in the high quality region discussed in section 4.3.4. The largest eigenvalue of the whole system increases due to the low quality cells, so the effectiveness of the smoothing in high quality regions is degraded. Even though local correction smoother is applied, it is still unable to recover the convergence rate to the satisfied level.

Applying the corrected eigenvalue technique described in section 4.3.4, the convergence of relative residual obtained by using the true largest eigenvalues and the corrected largest eigenvalues inputs is shown in fig. 4.21. The convergence rate improves more if using the corrected largest eigenvalues in Chebyshev smoother, and reaches the reference level



(a) P_1 element (b) P_2 element

Fig. 4.20 Relative residual obtained by multigrid with Chebyshev smoother using the largest eigenvalues of the whole system input, carried out on the unit cube meshes with low quality regions on all levels.



(a) P_1 element (b) P_2 element

Fig. 4.21 Relative residual obtained by multigrid with Chebyshev smoother using the corrected eigenvalues working on the unit cube meshes.

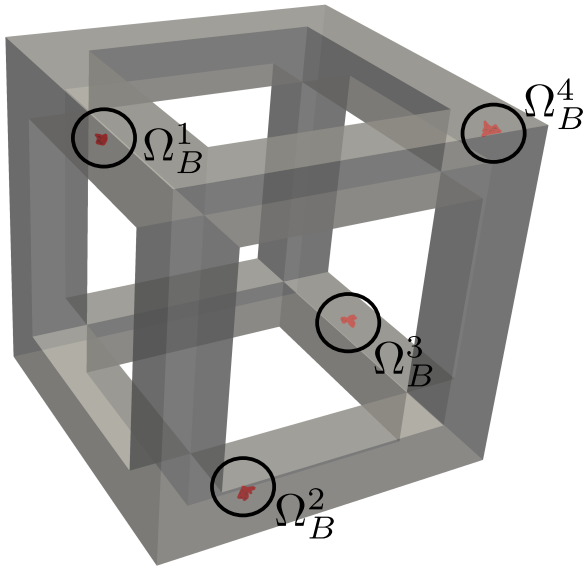


Fig. 4.22 The geometry of the lattice and the positions of the low quality cells on the finest grid.

similar to the Gauss–Seidel case. Therefore, the $O(n)$ property of multigrid with Chebyshev smoother is achieved with respect to meshes of different mesh qualities.

4.5.3 Linear elasticity on a lattice

The domain tested here is a truss-like lattice which can be served as an analytical model for the solid beam theory. The geometry is defined by

$$\begin{aligned}\Omega &= a^3 \setminus (a \times b^2 \cup b \times a \times b \cup b^2 \times a),\\ a &= [0,6],\\ b &= [1,5],\end{aligned}$$

and shown in fig. 4.22. Four-level hierarchy grids are provided. Locally poor quality cells are randomly distributed on each level. The positions of the low quality regions on the finest grid are also shown in fig. 4.22. The linear elasticity equation is solved here with

$$-\operatorname{div}\sigma(u) = f \quad \text{in} \quad \Omega, \tag{4.22}$$

level	$\gamma_{\min}(\Omega_{\text{high}})$	$\gamma_{\min}(\Omega_{\text{low}})$	number of cells in Ω	number of cells in Ω_B	number of DOFs in Ω P_1/P_2	number of DOFs in Ω_B P_1/P_2
1 (fine)	0.209	2.64×10^{-8}	711.683	732	437 946/3 192 258	825/4506
2	0.228	2.7×10^{-8}	78 100	585	58 992/391 767	798/4101
3	0.250	6.43×10^{-8}	8341	444	8292/49 131	612/3027
4 (coarse)	0.344	1.07×10^{-6}	1080	96	1284/7116	183/816

Table 4.5 Mesh quality and problem sizes of the lattice mesh hierarchies.

where u is the displacement field and $\sigma(u)$ is the stress tensor satisfying the isotropic elastic law

$$\sigma(u) = 2\mu\varepsilon(u) + \lambda \operatorname{tr}(\varepsilon(u))I, \tag{4.23}$$

where the deformation $\varepsilon(u)$ relating the displacement filed u by

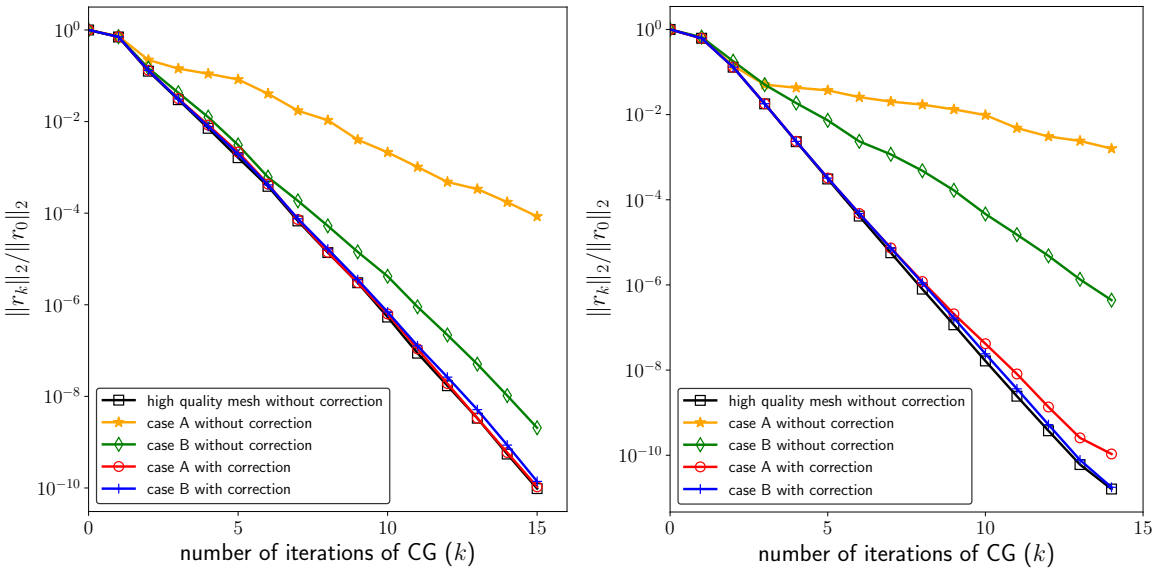
$$\varepsilon(u) = \frac{1}{2}(\operatorname{grad}(u) + \operatorname{grad}(u)^T), \tag{4.24}$$

and μ and λ are the Lamé constants based on the material property. For this problem, we use the property of Aluminium with Young's modulus 69 GPa and the Poisson's ratio 0.33. Suppose there is no body force, i.e. $f = (0,0,0)$. The boundary conditions are given by

$$\begin{aligned} u &= (0,0,0) \quad \text{on} \quad \Gamma_1 = \{(x,y,z) \in \partial\Omega : x = 0\} \\ \sigma(u)\cdot n = T &= (10^3,0,0) \quad \text{on} \quad \Gamma_2 = \{(x,y,z) \in \partial\Omega : x = 6\}. \end{aligned}$$

The details including the mesh quality and mesh sizes of levels are listed in table 4.5. It is clear that the low quality regions only take up a small portion of the whole domain. However, it is found that only these low quality cells can cause significant problem.

Multigrid preconditioned conjugate gradient (CG) is carried out with one V-cycle applied for each iteration of CG. Symmetric Gauss–Seidel is served as the global smoother. The results of relative residual with cycle counts obtained in both cases are plotted in fig. 4.23 for P_1 and P_2 elements. The convergence rate of the multigrid preconditioned CG method also decays dramatically for low quality meshes. The high frequency error remaining in low quality regions can not be removed by the multigrid, nor by the conjugate gradient method, and would also persist through CG iterations. For the case B that low quality cells only appear on intermediate grids, the convergence rate is still acceptable for P_1 element, whereas is much slower for P_2 element. Apparently, the low quality cells have larger effects on the P_2 element, which is commonly employed in engineering applications. The convergence on the low quality meshes can be fully recovered by the proposed smoother. Therefore, if the



(a) P_1 element (b) P_2 element

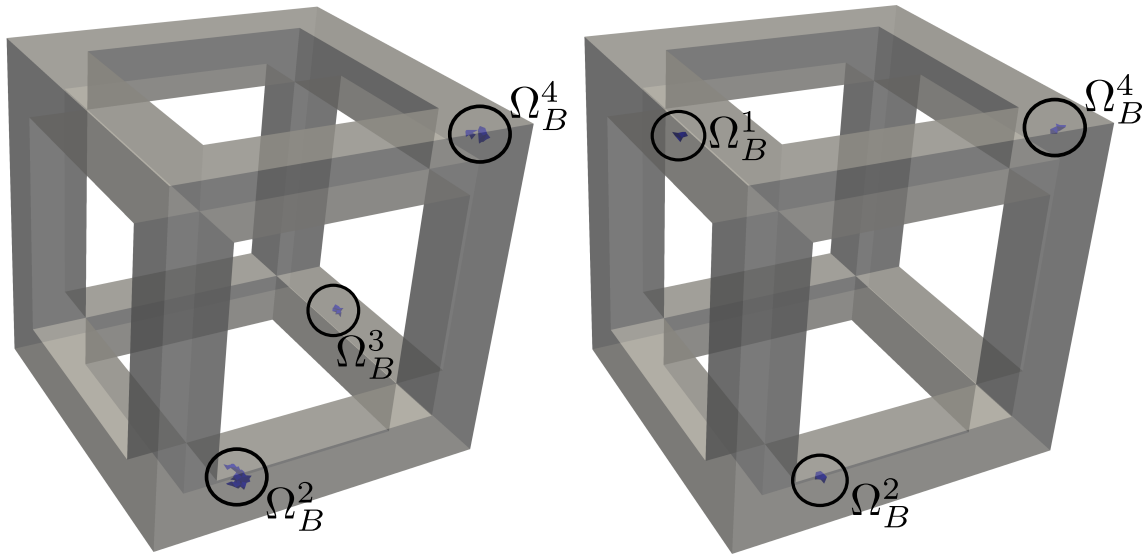
Fig. 4.23 Relative residual obtained by multigrid preconditioned conjugate gradient (CG) with and without local correction working on the lattice meshes with low quality regions on all levels (case A) and with low quality regions on all levels except the finest level (case B).

high frequency error can be eliminated in the multigrid cycles, then there is no concern to use multigrid preconditioned Krylov solver.

Figure 4.24 gives the places on the finest grid with large value of the residual after ten iterations of multigrid preconditioned CG for the case of low quality cells on all levels and with local correction on coarse grids only. The large value residual areas still collapse the regions containing locally poor quality cells. It validates that the high frequency error produced by multigrid smoother on low quality cells also persists through the conjugate gradient iterations. These error can be damped out completely by the local correction smoother. The example shows the effectiveness of the proposed smoother for the multigrid preconditioner, and its capability to solve vector-valued elasticity problems.

4.5.4 Linear elasticity on a dumbbell–like structure

Finally, we consider the linear elasticity problem on a dumbbell-like structure with the geometry containing two hexagons connected by three slender bars, shown in fig. 4.25. The material properties of Aluminium is still used. The left and right boundaries of the geometry are pulled by a uniform horizontal force of magnitude 10^3, and a uniform load of $f = (0, 0, -10^3)$ is also applied. Four level mesh hierarchies are provided. In this particular problem, different mesh levels are used to fit the problem sizes of P_1 and P_2 elements. The

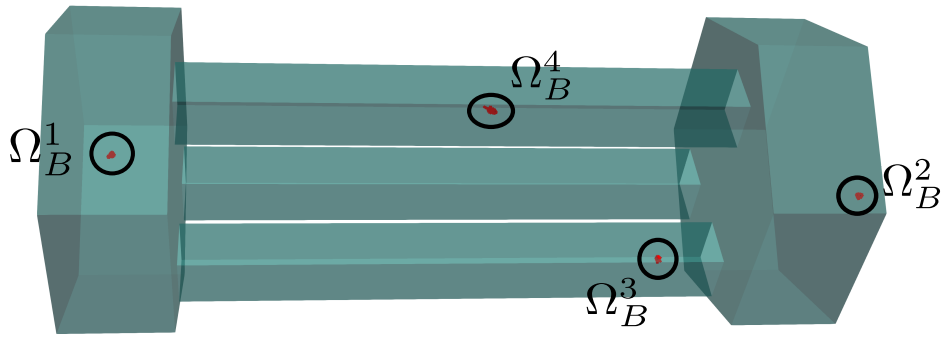


(a) P_1 element (b) P_2 element

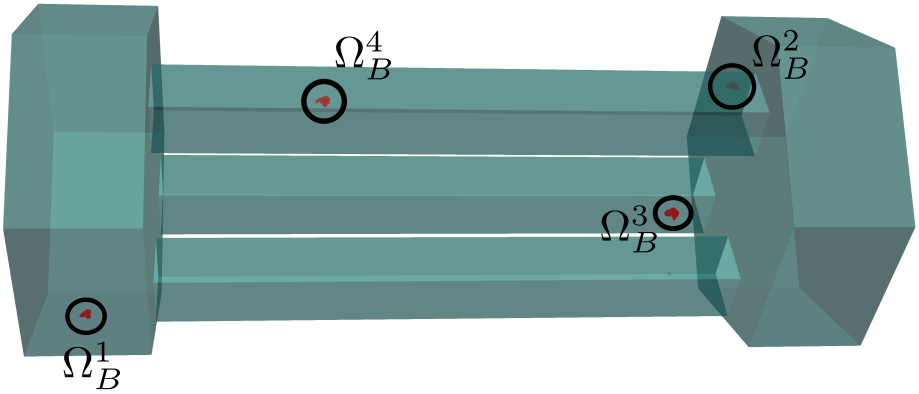
Fig. 4.24 Positions on the finest lattice mesh holding large value of residual after ten iterations of multigrid preconditioned CG.

low quality regions on the finest grids for P_1 and P_2 elements have been shown in fig. 4.25. The information of the mesh quality and the problem sizes of different mesh levels is given in table 4.6. It is noted that, in this example, the coarsening factors in terms of number of DOFs between levels are different. The coarsening is much aggressive between the simulation level and the level two, which reaches a factor of three reduction in the average mesh size. For the coarse levels, the coarsening becomes standard, e.g. a factor of two reduction in average mesh size. This set-up is commonly used in engineering applications when the mesh of the simulation(finest) level is relatively large. A large coarsening between the first and the second level can significantly decrease the computational cost in the context of using multigrid. On the other hand, this configuration fits well the case B, in which the coarse grids are just generated to accelerate the calculations, so that relatively small meshes are desired for coarse grids.

We apply the Jacobi preconditioned Chebyshev smoother for this problem. Again, the multigrid preconditioned conjugate gradient (CG) method is carried out. The convergence results obtained with and without local correction and with the corrected largest eigenvalue strategy used in the Chebyshev smoother are shown in fig. 4.26. It is clear that the convergence is much slower if using any low quality mesh for both cases. And the performance of the solver is much worse when there are poor quality cells on the simulation level. It is found that if only using low quality intermediate grids to accelerate the calculations, the convergence rate for P_2 element can still be dramatically slow and unacceptable.



(a) Level 1 for P_1 element

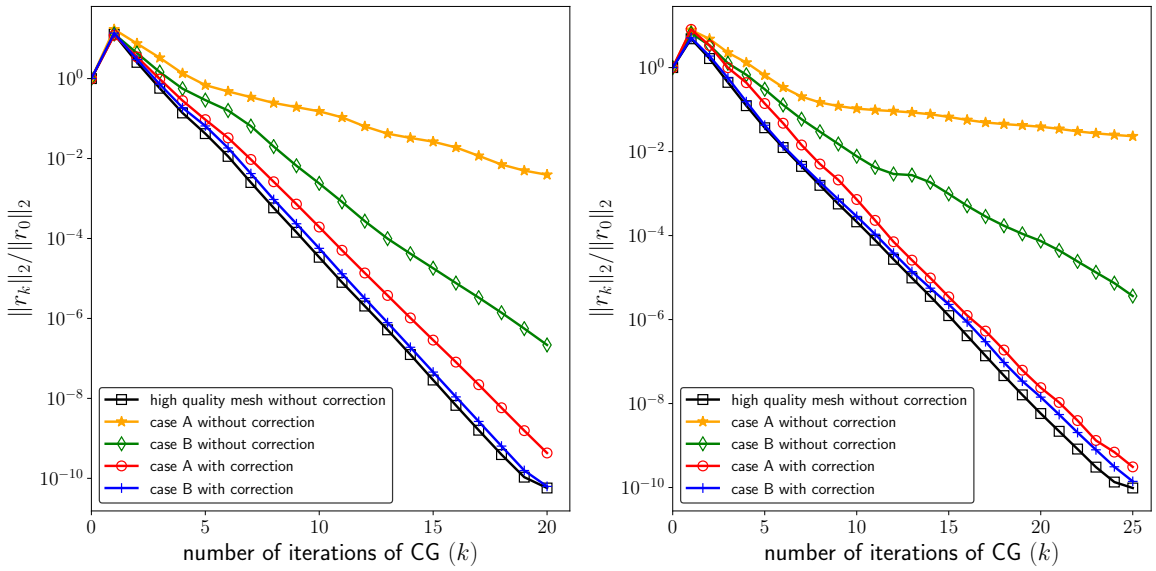


(b) Level 1 for P_2 element

Fig. 4.25 The geometry of the dumbbell–like structure and the positions of locally poor quality cells on the finest grid of the P_1 and P_2 elements.

P_1 element							
level	cell size	$\gamma_{\min}(\Omega_{\text{high}})$	$\gamma_{\min}(\Omega_{\text{low}})$	number of cells in Ω	number of cells in Ω_B	number of DOFs in Ω	number of DOFs in Ω_B
1 (fine)	0.025	0.150	3.18×10^{-8}	3982354	788	2130582	876
2	0.08	0.204	6.94×10^{-8}	139469	538	90027	699
3	0.2	0.273	2.93×10^{-8}	12627	454	10455	615
4 (coarse)	0.5	0.179	8.92×10^{-8}	1345	164	1515	216
P_2 element							
level	cell size	$\gamma_{\min}(\Omega_{\text{high}})$	$\gamma_{\min}(\Omega_{\text{low}})$	number of cells in Ω	number of cells in Ω_B	number of DOFs in Ω	number of DOFs in Ω_B
1 (fine)	0.04	0.220	3.26×10^{-8}	1035068	600	4453713	3771
2	0.12	0.211	6.27×10^{-8}	48880	381	237942	2664
3	0.25	0.267	8.34×10^{-8}	5059	253	28863	1926
4 (coarse)	0.5	0.179	8.92×10^{-8}	1345	164	8532	1116

Table 4.6 Mesh quality and problem sizes of the dumbbell–like structure mesh hierarchies.



(a) P_1 element (b) P_2 element

Fig. 4.26 Relative residual obtained by multigrid preconditioned conjugate gradient (CG) on the dumbbell–like structure using the Chebyshev smoother with corrected eigenvalues input, for low quality regions on all levels (case A) and for low quality regions on all levels except the finest level (case B).

The detailed number of iterations of multigrid preconditioned CG needed to reduce the relative residual smaller than 10^{-10} is shown in table 4.7, especially for using the true/default and corrected largest eigenvalues $\lambda_{\max}$ for the Chebyshev smoothers. It is observed that if the low quality finest grid is used (case A), the convergence stalls for both elements. If the poor quality cells appear on the intermediate grids only (case B), the computational time increases by a factor of 1.5 to 2 compared to the reference solution. For a million size problem, the extra computational cost can be significant, especially for P_2 element. The local correction scheme improves the convergence rate in both cases. The maximum eigenvalue inputs in Chebyshev smoother do influence the convergence. It is not that remarkably slow if using the largest eigenvalues of the whole system, while the proposed corrected largest eigenvalue technique can save around $20-30\%$ of the computational time. A multigrid preconditioner with the Chebyshev smoother using the corrected largest eigenvalue can restore the convergence rate on the low quality mesh to the reference level.

In terms of time cost, we consider three components of the calculation: (i) construction of FEM system, (ii) multigrid set–up including constructions of coarse grid systems and transfer operators, tracking low quality cells, and making local subsystems, and (iii) multigrid preconditioned CG iterations. The detailed time cost in seconds for relative residual to reach

element type	P_1 element		P_2 element	
case	case A	case B	case A	case B
high quality mesh	20	20	25	25
low quality mesh without correction	~ 100	30	> 100	49
low quality mesh with correction using true $\lambda_{\max}$ in Chebyshev	32	25	45	39
low quality mesh with correction using corrected $\lambda_{\max}$ in Chebyshev	22	20	27	26

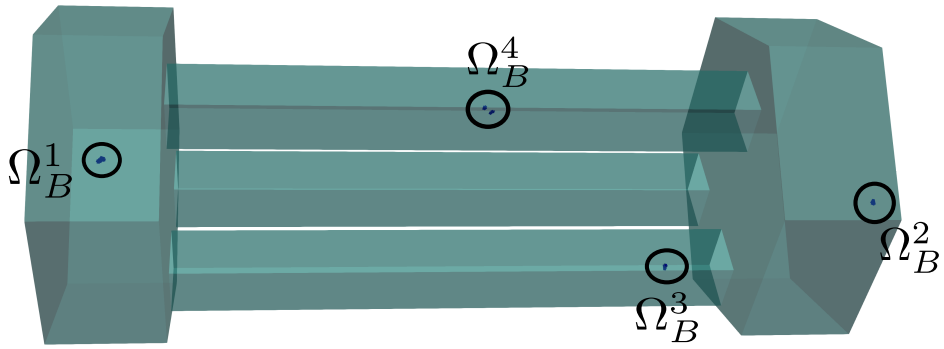
Table 4.7 Number of iterations of multigrid preconditioned CG with Chebyshev smoother needed for relative residual to reach 10^{-10} for low quality regions on all levels (case A) and for low quality regions on all levels except the finest level (case B).

Element type	Case	FEM set-up (s)	Multigrid set-up (s)	CG iterations (s)
P_1 element	High quality mesh	56.67	28.38	103.41
	Low quality mesh	55.81	27.85	564.24
	Case A with correction	56.72	34.12	155.42
	Case B with correction	56.28	30.77	126.96
P_2 element	High quality mesh	87.91	218.51	1233.61
	Low quality mesh	88.14	218.61	-
	Case A with correction	87.86	250.07	1625.97
	Case B with correction	88.11	232.18	1352.75

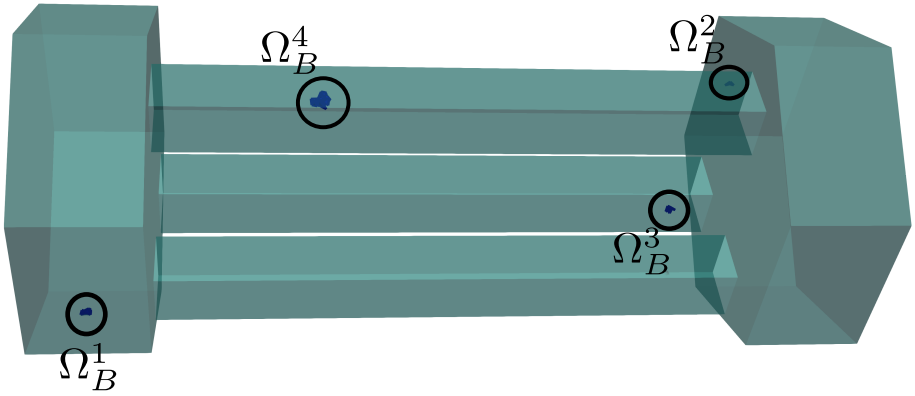
Table 4.8 Computational time in seconds for solving the dumbbell-like structure problem, in which there are three components: (i) FEM system construction, (ii) multigrid set–up including constructing coarse grid systems and transfer operators, tracking low quality cells, and making local correction systems, and (iii) multigrid preconditioned CG iterations.

10^{-10} is listed in table 4.8. It is clear that the computational time for low quality mesh without local correction can be extremely high, and for P_2 element, the convergence stops at relative residual of 10^{-3}. With local correction, though more time is spent, the convergence is recovered in reasonable time. It is shown that the time cost for multigrid set–up does not increase too much if low quality cells only occur on the intermediate grids (Case B). It is observed that the multigrid with local correction takes about $30-40\%$ of the overall computational time.

Finally, in fig. 4.27, we show the areas containing large value of residual on the finest grid after ten iterations of multigrid preconditioned CG with local correction on the coarse grids only. The positions where the residual is large collapse in the regions of the low quality cells, which verifies that the smoothing is damaged locally in low quality regions. The local correction resolves the poor smoothing in low quality regions, and recovers the



(a) P_1 element



(b) P_2 element

Fig. 4.27 Areas containing large value of the residual on the finest dumbbell–like structure mesh of P_1 and P_2 elements after ten iterations of multigrid preconditioned CG with local correction on coarse grids only.

convergence rate. The example shows that the method is robust for large size problems and large coarsening factors, which can be appealing in the real–world applications.

4.6 Concluding remarks

It is observed that multigrid converges slowly with meshes containing local regions of poor quality cells. The slow convergence is due to the local failure of the smoothing property. High frequency errors in low quality regions are hard to be eliminated by the smoothers, and persist through multigrid cycles.

A global–local combined smoother is developed for geometric multigrid to deal with problems with a small number of low quality cells. The global–local smoother is made up of a global smoother on the whole domain and a local correction smoother on the subdomains of poor quality cells. It is shown that this combined smoother is equivalent to a full overlapped Schwarz type domain decomposition method, and in the form of a block Gauss–Seidel.

Numerical examples are carried out to validate the performance of the new smoother. The local correction smoother can remove the high frequency error remaining in the area of low quality cells. Remarkably, it is shown that the poor convergence rate on the low quality mesh can be recovered to the reference level of using the high quality mesh. The multigrid with local correction scheme opens up the possibility of high efficiency multigrid to work with extreme scale complicated engineering simulations in system level.

Chapter 5

Local correction for classical AMG

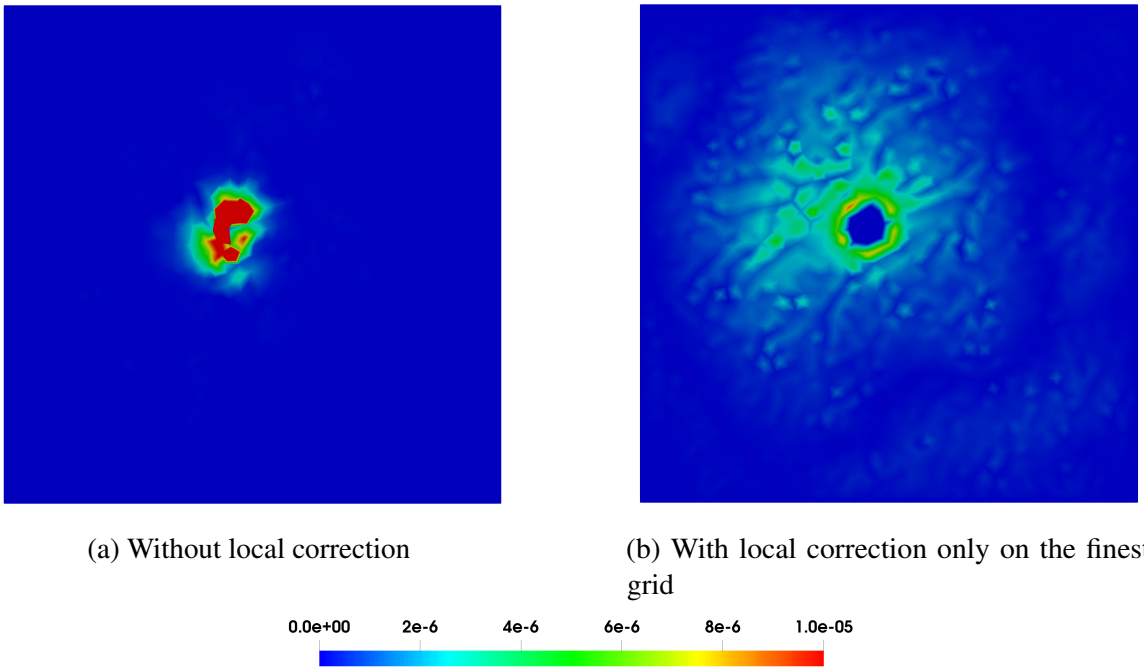
For complicated engineering problems, geometric mesh hierarchies are usually not available, and the algebraic multigrid (AMG) is more attractive. The convergence of AMG also suffers from the presence of low quality cells [93], therefore finding the way to apply the local correction for the AMG setting is of the great value to deal with complex engineering applications.

In this chapter, we discuss the use of the local correction scheme for the classical AMG which is introduced in section 3.6.1. It is found that if only applying the local correction on the finest grid of the classical AMG, the convergence rate is still not fully recovered. We propose a way to track the 'low quality regions' on the abstract coarse grids of the classical AMG via the information transfer between levels. With the help of local correction on the abstract coarse grids, the converge rate of using the classical AMG on low quality meshes can be restored to the reference level, which is desired.

A model problem is presented in section 5.1 to illustrate two facts: (i) The local correction on the finest grid is unable to restore the convergence rate of classical AMG on the low quality mesh to the reference level of using the high quality mesh; and (ii) There is more high frequency error propagating outward the low quality regions to the neighbouring cells on the abstract coarse grids. We propose an algorithm to find the 'low quality regions' on the abstract coarse grids in section 5.2. In section 5.3, several numerical experiments are demonstrated to validate the proposed algorithm. Some conclusions are given in section 5.4.

5.1 Problems in classical AMG with low quality cells

In the AMG setting, the local correction smoother can easily be applied on the finest grid (simulation level). The main question becomes whether this is sufficient to obtain a satisfactory convergence rate, i.e., a full recovery of convergence rate to the level of using



(a) Without local correction

(b) With local correction only on the finest grid

Fig. 5.1 Contour plots of the absolute value of residual on the low quality unit square after ten cycles of classical AMG with (a) no local correction and (b) local correction on the finest grid.

high quality meshes. We first use a model problem to show that only applying local correction on the finest grid is not enough to obtain a full recovery of convergence rate, and provide an explanation of why.

5.1.1 A model problem

The unit square mesh shown in fig. 4.2 with the low quality region at the centre is still used here. We test the homogeneous Poisson's equation with homogeneous boundary conditions. Three-level classical AMG is carried out, and the symmetric Gauss-Seidel method is chosen as the smoother. The contour plots of the absolute value of residual on each vertex obtained after ten cycles of the classical AMG without local correction, and with local correction only on the finest grid are shown in fig. 5.1. Clearly, without local correction, high frequency error still persists in the region of low quality cells, which supports that the weak smoothing due to the low quality cells also happens in classical AMG. If we apply the local correction only on the finest grid, parts of high frequency error can be removed in the low quality region. However, it can be seen that the high frequency error propagates outward from the region of

low quality cells to the neighbouring cells surrounding the low quality region. This extended high frequency error is hard to be eliminated by the local correction only on the finest grid.

5.1.2 High frequency error on the abstract coarse grid

Based on our experience with GMG, applying the local correction on the finest grid can still help improve the convergence of the classical AMG since it is able to remove high frequency error in the local correction region Ω_B. However, in the model problem, there is more high frequency error produced, which is outside the local correction region on the finest grid, and it is found that these exceeded error can not be eliminated by the finest grid correction.

A naive way to fix the problem is to extend the original low quality region Ω_B, and hope that all error propagating outside Ω_B can be removed completely on the finest grid. However, this way is not desirable. The major drawback is that it is difficult to predict to what extent we need to extend the low quality region on the finest grid to make sure that areas containing the exceeded high frequency error can all be covered. Another disadvantage is that a large extension of low quality cells on the finest grid (simulation level) can substantially enlarge the size of the local correction system, which may lead to much higher computational costs.

If the high frequency error can not be removed by the local correction on the finest grid, then it is suggested that these error may be generated on the algebraic coarse grids in the classical AMG. Denote the high frequency error generated by the low quality cells on the level l by e_l. We can then decompose the high frequency error e_{tot} due to the low quality cells by

$$e_{tot} = e_1 + e_2 + \cdots + e_L. \tag{5.1}$$

It is observed from the model problem that the error on the finest grid dominates the total high frequency error, namely,

$$\|e_{tot}\| \sim \|e_1\| \gg \|e_2 + \cdots e_L\|. \tag{5.2}$$

Local correction on the finest grid removes the error e_1, thus, the extended error can be regarded as e_2, e_3, $\cdots$, and e_L. We have already seen, in GMG case, that local correction can remove the high frequency error on coarse grids, then the key becomes to track the 'low quality regions' on the abstract coarse grids of classical AMG.

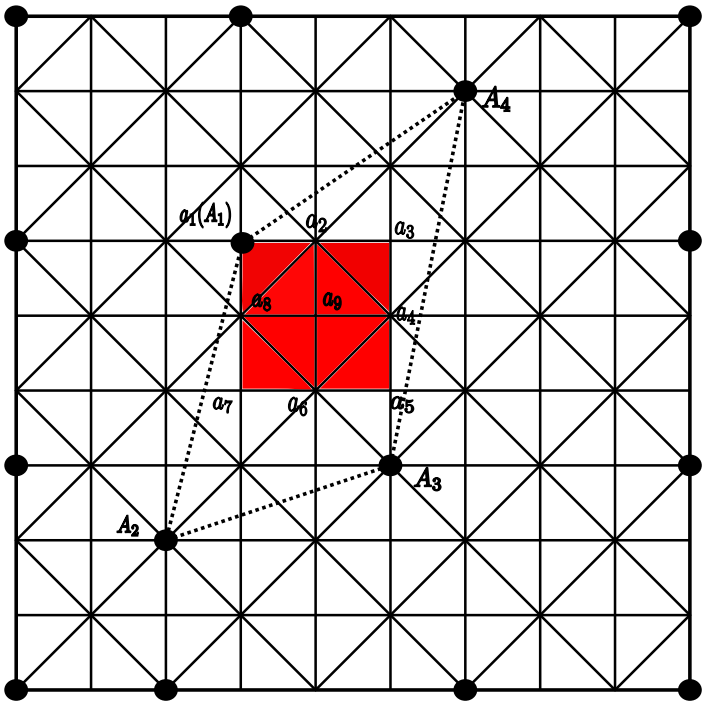


Fig. 5.2 A unit square mesh showing the low quality regions on the abstract coarse grid of the classical AMG, in which the unit square mesh is the fine grid, those bold nodes are the C nodes selected by the C/F splitting, and the coloured region stands for the low quality region on the fine grid.

5.2 Identifying low quality regions on the abstract coarse grid

To find the low quality regions on the abstract coarse grid, let us consider a two–level system. Denote the low quality region on the fine grid and on the coarse grid by Ω_B^f and Ω_B^c respectively. As discussed above, the region containing high frequency error on the coarse grid propagates outside the low quality region on the fine grid, so it holds that $\Omega_B^f \subset \Omega_B^c$. We aim to track the extended region of the high frequency error $\Omega_B^c \setminus \Omega_B^f$. If Ω_B^f is determined, we propose a way to relate the DOFs in Ω_B^f to the DOFs in Ω_B^c.

A structured unit square mesh, shown in fig. 5.2, is used to illustrate the idea. Recall that the coarse grid in the classical AMG is constructed by the C/F splitting, which is explained in section 3.6.1. Suppose that the original unit square mesh is the fine grid, and that the bold nodes on the mesh are the C-nodes selected as the coarse grid nodes by C/F splitting. Let the region coloured red be the low quality region on the fine grid, i.e. Ω_B^f, with vertices $a_1, a_2, \cdots, a_9$. The interpolatory set K, defined in the direct prolongation operator eq. (3.32),

for these nodes are set to be A_1, A_2, A_3, and A_4, namely,

$$\begin{aligned} a_1 &= A_1, \\ A_1, A_2, A_3, A_4 &\in K_{\{a_1, a_2, \cdots, a_9\}}. \end{aligned} \tag{5.3}$$

By the restriction of classical AMG, the high frequency error in the low quality region Ω_B^f of the fine grid will be transferred to the region with vertices A_1, A_2, A_3, and A_4 on the coarse grid, i.e., the area enclosed in the dashed line, and we define this area as the low quality region on the abstract coarse grid Ω_B^c. Apparently, even applying the local correction on the fine grid low quality region Ω_B^f, the high frequency error can not be completely removed due to $\Omega_B^f \setminus \Omega_B^c \neq \emptyset$. The essence of this connection is that there is an information transfer between Ω_B^f and Ω_B^c by the transfer operators.

In order to find the abstract area of Ω_B^c on the abstract coarse grid, we aim to find the DOF set of this region. Denote the DOFs in Ω_B^f and Ω_B^c by N_B^f and N_B^c respectively. Suppose that the node $i \in F$ is one of the nodes in the fine grid low quality region, i.e. $i \in N_B^f$, then two situations happen. First, if the node i also appears on the coarse grid, i.e. $i \in C$, then we definitely include this node to the coarse grid low quality DOF set N_B^c. Second, if the node $i \in F$ is not on the coarse grid, then the question becomes to find those nodes on the coarse grid that have an information transfer with the node i on the fine grid. We focus on the prolongation operator in eq. (3.32). For the node $i \in F$, there exists some j (can be several) such that $j \in C$ with $j \in K_i$, and $P_{ij} \neq 0$. Then, there is an information transfer between the node $i \in F$ and the node $j \in C$, and we include the node j in N_B^c. We summarise the idea in the following,

$$i \in N_B^f : \begin{cases} i \in C, & i \to N_B^c, \\ i \in F, & \exists j \in C, \text{ such that } j \in K_i, \quad j \to N_B^c. \end{cases}$$

In the example of fig. 5.2, the nodes in the low quality region on the fine grid is $N_B^f = \{a_1, a_2, \cdots, a_9\}$, and the resulting DOF set of the coarse grid low quality region becomes $N_B^c = \{A_1, A_2, A_3, A_4\}$. The procedure to track the low quality regions on the algebraic coarse grid is illustrated in algorithm 5.1. After the low quality DOF set on coarse grid is obtained, then we just insert the DOF set N_B^c to eq. (4.6) and eq. (4.7) to construct the local correction system on the coarse grid.

Algorithm 5.1 Track the low quality region on the coarse grid of classical AMG.

1: Find the regions of the low quality cells on the fine grid Ω_B^f and its corresponding DOF set N_B^f.
2: Set the DOF set of the coarse grid low quality region $N_B^c = \emptyset$.
3: **for** $i \in N_B^f$. **do**
4: **if** $i \in C$ **then**
5: $N_B^c = N_B^c \cup \{i\}$.
6: **else**
7: $i \in F$, find $j \in C$ such that $j \in K_i$.
8: [for implementation: this is equivalent to find $j \in C$ such that $P_{ij} \neq 0$.]
9: $N_B^c = N_B^c \cup \{j\}$.

5.3 Numerical results

The performance of the proposed scheme is examined here through several numerical examples from simple to complicated ones. Lagrange elements are used with linear P_1 and quadratic P_2 bases on simplices. The direct prolongation in eq. (3.32) for the classical AMG is employed. The global smoothers used here are the symmetric Gauss–Seidel and the Jacobi preconditioned Chebyshev methods, which are applied twice in terms of pre- and post-smoothing. A direct solver is used on the coarsest grid. The coarsening parameter defined in the strongly coupling for the classical AMG (see eq. (3.27)) is chosen as 0.15 for 2D and 0.08 for 3D meshes. The scalar Poisson equation is tested and the solver is terminated once the relative residual reaches 10^{-12} measured in the 2-norm.

The finest grids are unstructured meshes with simplex cells, which are created using the Gmsh [48]. Again, we move several vertices in the generated high quality meshes to control the mesh quality. The low quality regions Ω_B on the fine grid are still determined by tracking the cells with $\gamma < 0.1$, where γ is the normalised radius ratio in eq. (4.1). Finite element calculations are implemented in FEniCS [5, 80, 79], and the C/F splitting and the transfer operators of the classical AMG are constructed using PETSc [11, 10, 12].

5.3.1 Model problem

The model problem in section 5.1.1 is solved again by the proposed local correction smoother on abstract coarse grids. The contour plot of the absolute value of the error on each vertex obtained by ten cycles of multigrid with local correction on all levels is shown in fig. 5.3, and we intentionally use the same scale in fig. 5.1 to make a comparison. The local correction on the abstract coarse grids successfully eliminates the extended error outside the fine grid low quality region. The reduction rate of relative residual with cycle number using the classical

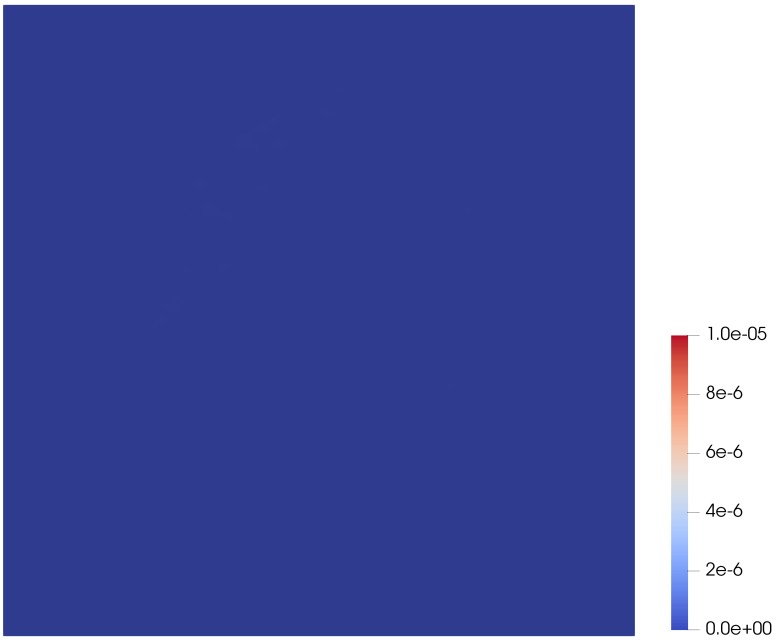


Fig. 5.3 Absolute value of the residual on each vertex of the unit square mesh after ten cycles of the classical AMG with local correction on all levels.

AMG with and without local correction is shown in fig. 5.4. Without applying any local correction, the convergence stagnates at the relative residual of 10^{-3}. If applying the local correction only on the finest grid, the convergence rate improves to the level of 10^{-6}, which supports our assertion that the error on the finest level dominates the total error. However, the convergence is still not fully recovered compared to the reference case. If using the proposed local correction on the abstract coarse grids, then a fully recovered convergence rate is achieved.

5.3.2 Poisson equation on a unit square

The second example is the unit square mesh shown in fig. 4.16 which contains three low quality regions with one touching the boundary. The Poisson equation is solved with

$$\begin{aligned} -\nabla^2 u &= 2\pi^2 \sin(\pi x)\sin(\pi y) \quad \text{in} \quad \Omega = (0,1)^2 \\ u &= 0 \quad \text{on} \quad \Gamma_D = \{(x,y) \in \partial\Omega : y = 0,1\}, \\ \frac{\partial u}{\partial n} &= \pi \sin(\pi y) \quad \text{on} \quad \Gamma_N = \{(x,y) \in \partial\Omega : x = 0,1\}. \end{aligned} \tag{5.4}$$

The finite element method with P_2 bases is tested. A four-level classical AMG is carried out with the coarsening factor in the classical AMG chosen as 0.15. The problem size of

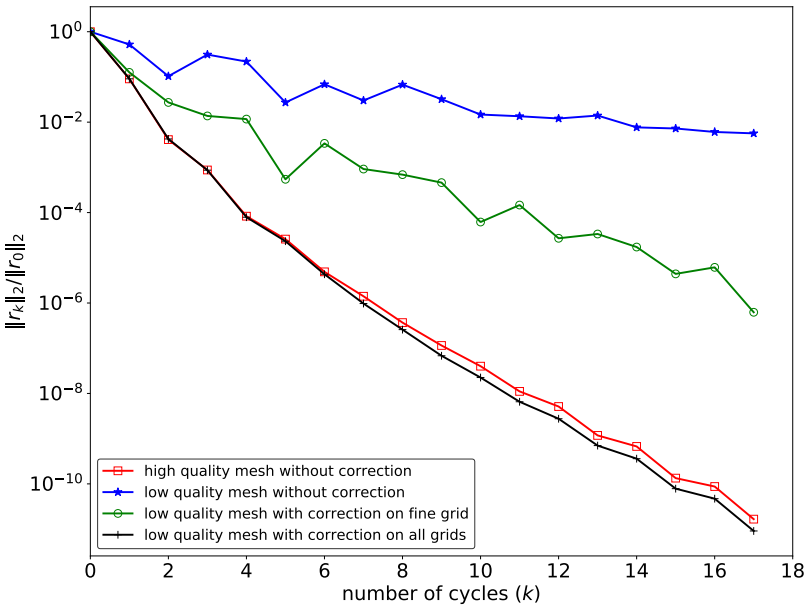


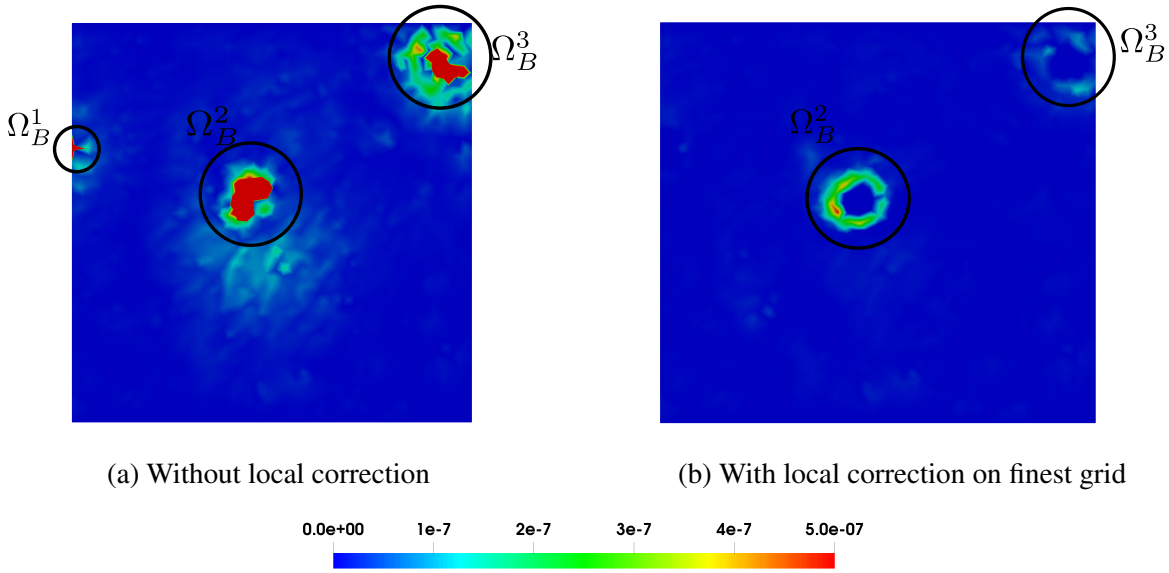
Fig. 5.4 Relative residual obtained by classical AMG without local correction and with local correction only on the finest grid and on all grids.

Element type	P_2			
Mesh level	1	2	3	4
Number of DOFs in Ω	8633	2257	574	102
Number of DOFs in Ω_B	134	89	60	-

Table 5.1 Problem size of the finite element systems and low quality regions on each level of the unit square problem.

the finite element system and the size of low quality region on each level of the classical AMG are listed in table 5.1. Contour plots of the absolute value of residual obtained by the classical AMG without local correction and with local correction only applied on the finest grid are shown in fig. 5.5. Similar phenomenon in the model problem is still obtained. Without local correction, the large value of residual clusters in regions of low quality cells. With local correction only applied on the finest grid, some parts of the high frequency error are removed, while large value residual persists in neighbouring cells of the low quality regions on the finest grid. These large value residual comes from the high frequency error propagating outside the low quality region on the finest grid as we predict.

After applying the local correction on the abstract coarse grids, the contour plot of the absolute value of residual on each vertex after ten cycles of the classical AMG is shown in fig. 5.6. Those large residuals on neighbouring cells are successfully damped out by the local correction on coarse grids. The plot of the relative residual with cycle number is shown



(a) Without local correction

(b) With local correction on finest grid

Fig. 5.5 Contour plots of the absolute value of residual on the finest unit square mesh after ten cycles of the classical AMG (a) without local correction and (b)with local correction on the finest grid only.

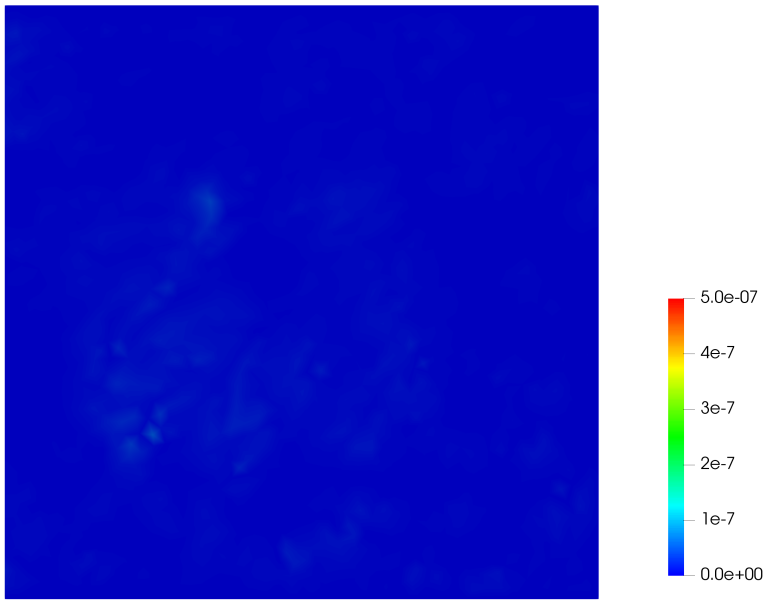


Fig. 5.6 Absolute value of residual on each vertex of the unit square after ten cycles of classical AMG with local correction on all grids

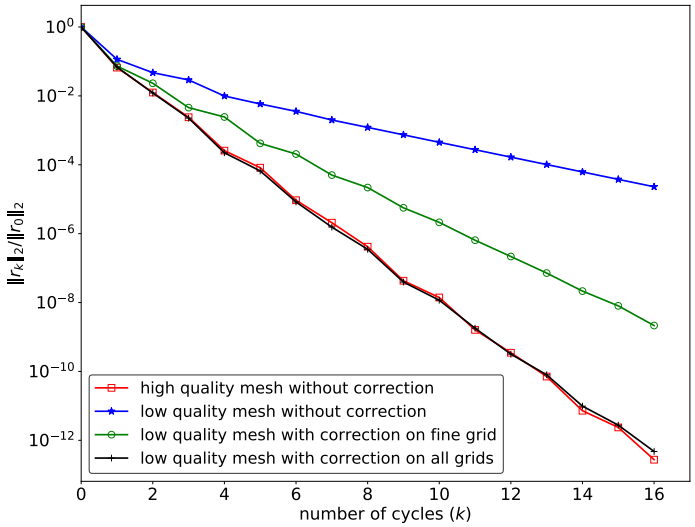


Fig. 5.7 Convergence rate obtained by the classical AMG with and without local correction on the unit square mesh.

in fig. 5.7. The convergence rate is similar to the model problem. The local correction on the finest grid improves the convergence rate, but is still unable to recover the convergence to the reference case. The observed convergence rate obtained by applying local correction on the finest grid is between the convergence rate obtained without local correction and the reference level. The proposed local correction on the abstract coarse grids gives a full recovery of the convergence rate.

5.3.3 Poisson equation on a unit cube

A unit cube mesh with four regions containing low quality cells is considered here and illustrated in fig. 5.8. The minimum normalised radius ratio is 0.218 for the high quality mesh and 1.83×10^{-6} for the low quality mesh. The Poisson equation with mixed boundary conditions is solved with

$$\begin{aligned} -\nabla^2 u &= 3\pi^2 \sin(\pi x)\sin(\pi y)\sin(\pi z) \quad \text{in} \quad \Omega = (0,1)^3, \\ u &= 0 \quad \text{on} \quad \Gamma_D = \{(x,y,z) \in \partial\Omega : x = 0,1 \quad \text{or} \quad z = 0,1\}, \\ \frac{\partial u}{\partial n} &= \pi \sin(\pi x)\sin(\pi z) \quad \text{on} \quad \Gamma_N = \{(x,y,z) \in \partial\Omega : y = 0,1\}. \end{aligned} \tag{5.5}$$

Both P_1 and P_2 elements are tested. The coarsening parameter for the classical AMG is taken as 0.08. A four-level classical AMG solver with the symmetric Gauss–Seidel as the global smoother is carried out. The number of total DOFs and the size of the local correction systems in different levels are reported in table 5.2.

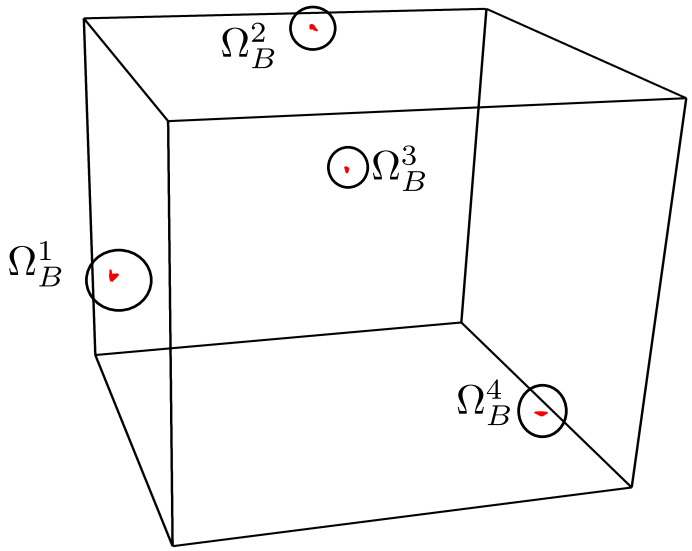
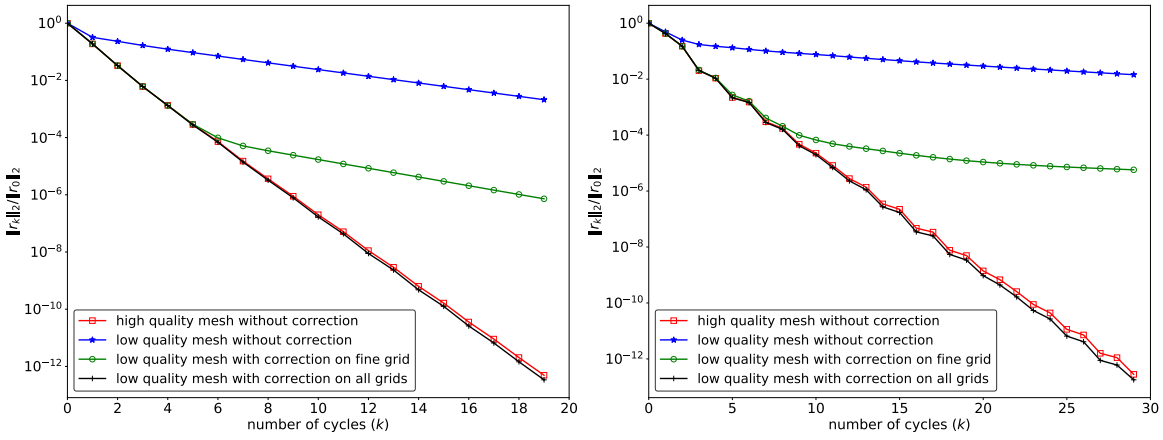


Fig. 5.8 A low quality unit cube mesh with four regions containing poor quality cells.

Mesh level	1	2	3	4
Element type		P_1		
Number of DOFs in Ω	45322	8004	1084	124
Number of DOFs in Ω_B	171	157	214	-
Element type		P_2		
Number of DOFs in Ω	344350	57633	8087	815
Number of DOFs in Ω_B	911	506	487	-

Table 5.2 Problem size of the finite element system and local correction system on each level of the unit cube problem.



(a) P_1 element (b) P_2 element

Fig. 5.9 Relative residual obtained by the classical AMG with and without local correction working on the unit cube mesh .

FEM error	$\|u_{15} - u^*\|_{L_2(\Omega)}$	
Element type	P_1	P_2
High quality mesh	9.19×10^{-4}	7.75×10^{-6}
Low quality mesh without correction	6.77×10^{-3}	8.19×10^{-5}
Low quality mesh with correction	2.24×10^{-3}	9.99×10^{-6}

Table 5.3 Finite element error in L_2 norm, u_{15} is the approximated solution after 15 multigrid cycles, and u^* is the exact solution to the PDE.

Figure 5.9 shows relative residual after each cycle for both P_1 and P_2 elements with and without local correction. The convergence rate is dramatically slow, or even stagnates if using the low quality mesh. If applying the local correction only on the finest grid, the relative residual decays rapidly to 10^{-4}, and then the decay becomes much slower. There is still a large difference compared to the reference case. If applying the local correction on the abstract coarse grids, the convergence rate matches the convergence rate of the high quality mesh.

We also show the exact finite element solution error in L_2 norm for both high and low quality meshes in table 5.3. Similarly, FEM error increases if using low quality mesh. And, with local correction, the error is reduced.

The example validates the use of local correction on abstract coarse grids of the classical AMG for a general 3D mesh.

5.3.4 Poisson equation on a lattice geometry

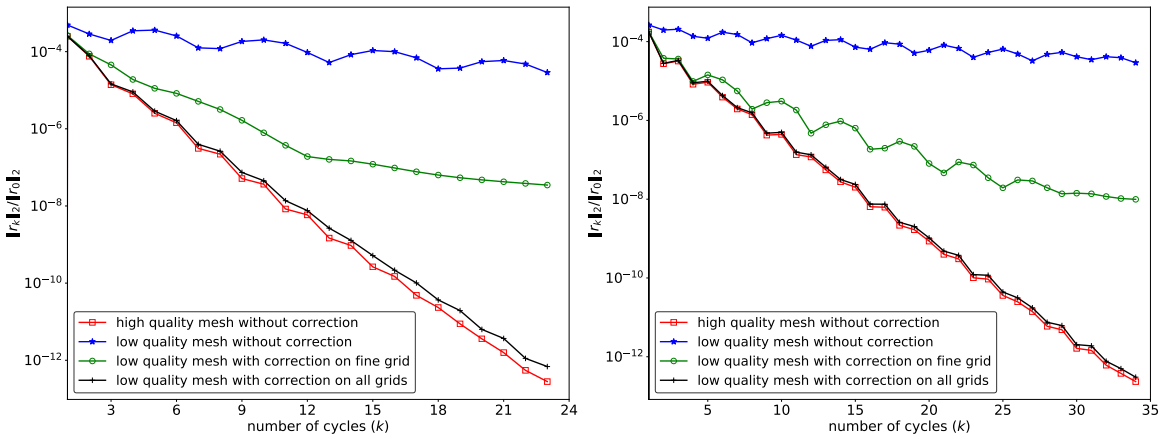
We consider a lattice geometry here with four regions containing low quality cells, which is shown in fig. 4.22. The smallest radius ratio is 0.21 for the high quality mesh and 1.19×10^{-7} for the low quality mesh. The Poisson problem is solved with

$$\begin{aligned} -\nabla^2 u &= 5 \quad \text{in} \quad \Omega, \\ u &= 1 \quad \text{on} \quad \Gamma_D^1 = \{(x,y,z) : z = 0\}, \\ u &= 2 \quad \text{on} \quad \Gamma_D^2 = \{(x,y,z) : z = 6\}, \\ \frac{\partial u}{\partial n} &= 0 \quad \text{on} \quad \partial\Omega \setminus (\Gamma_D^1 \cup \Gamma_D^2). \end{aligned} \tag{5.6}$$

A four-level classical AMG solver is carried out for both P_1 and P_2 elements. The numbers of DOFs and the sizes of local correction systems on all levels are shown in table 5.4. First, the symmetric Gauss–Seidel is taken as the global smoother. The convergence of relative residual

Mesh level	1	2	3	4
Element type		P_1		
Number of DOFs in Ω	145982	30249	4311	497
Number of DOFs in Ω_B	275	251	274	-
Element type		P_2		
Number of DOFs in Ω	1064086	189037	30538	3979
Number of DOFs in Ω_B	1502	780	718	-

Table 5.4 Problem size of the finite element system and local correction system on each level of the lattice problem.



(a) P_1 element (b) P_2 element

Fig. 5.10 Relative residual obtained by classical AMG with and without local correction working on the lattice example.

with cycle counts is shown in fig. 5.10. The results are similar to previous examples. For both P_1 and P_2 elements, the convergence completely stagnates for the low quality mesh without using the local correction. The local correction on the finest grid improves the convergence rate, and the relative residual decays to 10^{-6}. The high frequency error on the finest grid extends to neighbouring cells of the low quality region and persists there, which can not be efficiently removed by the finest grid correction, thus the relative residual stagnates at 10^{-8}. The local correction on the abstract coarse grids recovers the convergence rate to the reference state of using the high quality mesh. The detailed computational time for the calculation is listed here. Again, we split the whole procedure into three components: FEM set–up, multigrid set–up, and multigrid cycles, similar to the example shown in section 4.5.4. For P_2 element, the convergence of multigrid completely stops and relative residual can not reach 10^{-10}. The computational time increases around 30% for carrying out the local

Element type	Mesh quality	FEM set–up	Multigrid set-up	Multigrid cycles
P_1	High quality mesh	4.58	6.26	34.83
	Low quality mesh	4.67	6.25	719.51
	Low quality mesh with correction	4.71	9.33	46.88
P_2	High quality mesh	14.86	51.31	121.06
	Low quality mesh	14.83	51.42	-
	Low quality mesh with correction	14.86	69.16	157.84

Table 5.5 Computational time in seconds for relative residual to reach 10^{-10} solving the lattice problem, in which there are three components: (i) FEM system construction, (ii) multigrid set–up including constructing coarse grid systems and transfer operators, tracking low quality cells, and making local correction systems, and (iii) multigrid cycles.

Level	1	2	3	4
Largest eigenvalue for high quality mesh	2.144268	2.041355	1.675609	1.554857
Largest eigenvalue for low quality mesh	3.931112	3.772201	7.177829	8.505380
Corrected largest eigenvalue for low quality mesh	2.144087	2.043796	1.670069	1.553076

Table 5.6 Spectral radius of the finite element system on each level of the classical AMG for construing the Chebyshev smoother.

correction procedure. Multigrid with local correction, though takes additional computational time, still converges in reasonable time.

On the other hand, we check the corrected largest eigenvalues technique for using the Chebyshev smoothers discussed in section 4.3.4. The spectral radius of the P_1 finite element systems for the high and low quality meshes on all levels are listed in table 5.6. The spectral radius increases for the low quality mesh due to the presence of the low quality cells in the grid. The increase in spectral radius occurs on all levels, which means that the low quality regions on the finest grid also have influence on the coarse levels. The relative residual after each cycle is shown in fig. 5.11 for the P_1 element problem using the Chebyshev smoother. We also list the number of the classical AMG cycles required for the relative residual to reach 10^{-10} in table 5.7.

Without the local correction, the residual decay for the low quality mesh stalls at 10^{-4}. If the local correction is only applied on the finest grid, the convergence slows down at 10^{-7}. It takes an unreasonably long time for the relative residual to reach 10^{-10} from 10^{-7}, approximately another 100 cycles. Our proposed local correction on the abstract coarse grids significantly improves the convergence rate. The corrected largest eigenvalue

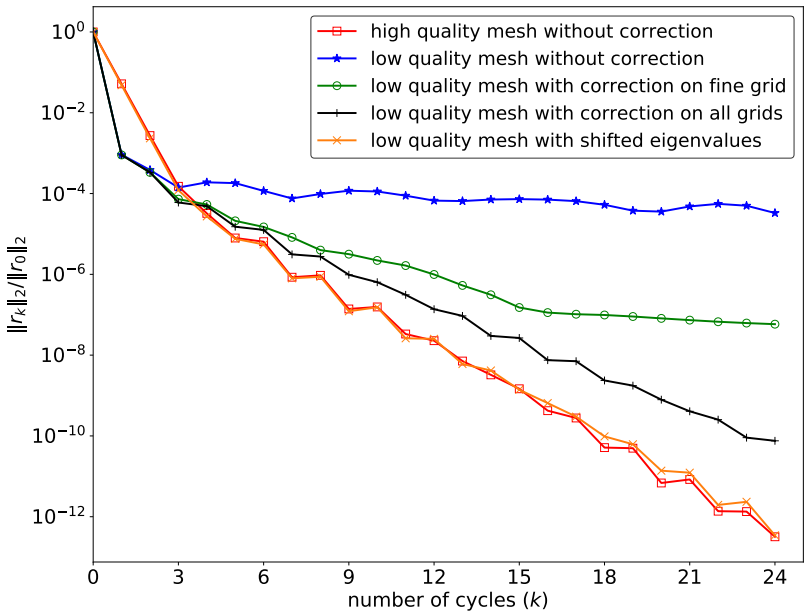


Fig. 5.11 Relative residual obtained by the classical AMG with Chebyshev smoother using different largest eigenvalue inputs for the P_1 element problem.

	Number of cycles
High quality mesh	18
Low quality mesh without local correction	>200
Low quality mesh with local correction on the finest grid	130
Low quality mesh with local correction on all grids	25
Low quality mesh with local correction on all grids and corrected eigenvalues in Chebyshev	18

Table 5.7 Number of the classical AMG cycles needed for the relative residual to reach 10^{-10} for the lattice example using the Chebyshev smoother.

technique for the Chebyshev smoother saves about 30% iteration counts compared to using the largest eigenvalue of the whole system. It validates that the corrected largest eigenvalue recovers the smoothing property of Chebyshev method in the high quality regions. Therefore, the convergence rate at the reference case can be finally achieved by the classical AMG with Chebyshev smoother using the corrected largest eigenvalues. The example shows the capability of our proposed idea of local correction for the classical AMG to solve real engineering problems.

5.4 Concluding remarks

The local correction has been successfully extended to the case of classical AMG. It is observed that the high frequency error due to the low quality cells propagates outward from the low quality region to the neighbouring cells. A way to find the appropriate local correction region on the abstract coarse grid is developed via the information transfer of low quality regions between levels. With the local correction applied on the abstract coarse grid, the poor convergence rate of the classical AMG on the low quality mesh can be restored to the reference level. This approach provides a platform to build a robust AMG solver for complex engineering simulations.

Chapter 6

Local correction for smoothed aggregation AMG

Smoothed aggregation AMG, introduced in section 3.6.2, is considered in this thesis to deal with vector–valued PDEs, such as linear elasticity. Smoothed aggregation AMG has already been served as the solver to solve the real-world complicated engineering problems [93]. However, a common view in engineering is that the smoothed aggregation AMG is not suitable for complex simulations, since there is a misconception that the smoothed aggregation AMG is a 'blackbox' method requiring no input or guidance other than the matrix to be solved. Actually, it is not the case, the proper use of the smoothed aggregation AMG needs several important inputs, e.g. the near-nullspace setting for 3D elasticity problems [46]. For low quality meshes, we assert that the largest eigenvalue used in constructing the smoothed prolongation is an important factor to control the convergence rate.

In this chapter, we discuss the use of local correction for smoothed aggregation AMG. It is found that the treatment used in the classical AMG does not lead to the full recovery of the convergence rate for smoothed aggregation AMG. The increase in the largest eigenvalue of the linear system due to the low quality cells degrades the performance of the smoothed prolongation in smoothed aggregation AMG. We propose a shifted largest eigenvalue technique, which shifts the largest eigenvalue back to a reasonable value to input. By the shifted eigenvalue strategy, the convergence rate of the smoothed aggregation AMG can be recovered to the reference level.

A summary of this chapter is as follows. A model problem is shown in section 6.1 to illustrate the problems of the increasing radius ratio used in the smoothed prolongation. The shifted eigenvalue technique is described in section 6.2. We demonstrate the performance of the method for a range of test problems in section 6.3. Finally, some concluding remarks are drawn in section 5.4.

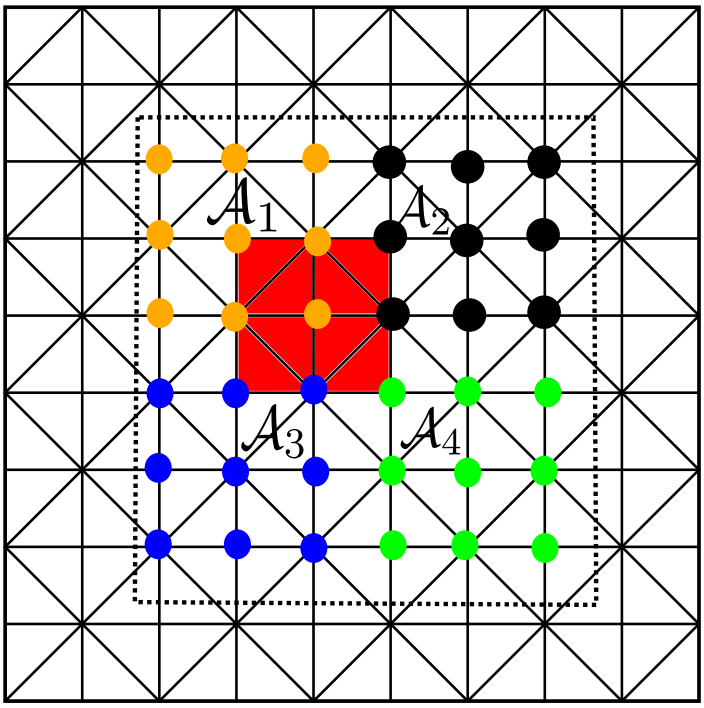
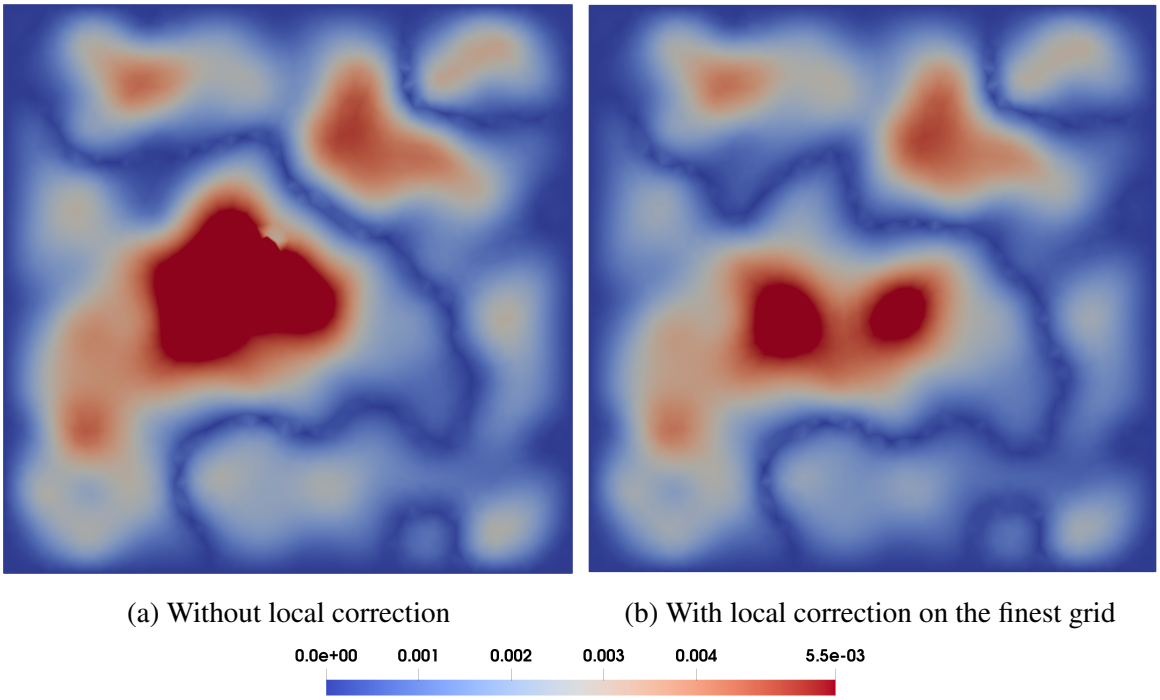


Fig. 6.1 An illustrated unit square mesh for smoothed aggregation AMG to track the low quality regions on the abstract coarse grid, in which the unit square mesh is the fine grid, bold coloured nodes stand for four aggregates, and the red region contains the low quality cells on the fine grid.

6.1 Issues with smoothed aggregation AMG

6.1.1 A model problem

The treatment similar to that in classical AMG of tracking the low quality regions on the abstract coarse grids can also be applied to smoothed aggregation AMG. To test, we use a similar structured uniform square mesh shown in fig. 6.1 to explain the expansion of low quality region on the coarse grid. Suppose that the unit square mesh is the fine grid, and that the region coloured red is the low quality region on the fine grid. For the smoothed aggregation AMG, the coarse grid is constructed by generating aggregates on the fine grid as discussed in section 3.6.2. Suppose that the low quality region is covered by four aggregates, $\mathscr{A}_1$, $\mathscr{A}_2$, $\mathscr{A}_3$, and $\mathscr{A}_4$, and each of them becomes a node/DOF on the coarse grid. The influence of the low quality region on the fine grid is then transferred to the region covered by the four aggregates on the coarse grid, i.e., the area inside the dashed line. Clearly, the low quality area becomes larger on the coarse grid. We can track this area in a similar manner to classical AMG using the transfer operators, as discussed in section 5.2.



(a) Without local correction (b) With local correction on the finest grid

Fig. 6.2 Contour plots of the absolute value of residual at each vertex on the unit square mesh after five cycles of smoothed aggregation AMG.

It is important to make a remark here. It is observed that the low quality region becomes very large on the abstract coarser levels for a large size problem. On the other hand, the local solution on the coarse levels does not affect the convergence too much in the smoothed aggregation AMG case, namely, the coarser the grid is, the less the influence on the final convergence rate becomes. As a result, the effects of the low quality regions are more significant on the finer levels of the smoothed aggregation AMG. We often apply the local correction only on the first two or three levels in the smoothed aggregation AMG cycles.

The model problem in section 5.1.1 is used here to examine the performance. Quadratic finite elements are applied, and the symmetric Gauss–Seidel is served as the global smoother. The coarsening factor is chosen as 0.15 in the definition of the strongly coupling in eq. (3.37). Four-level smoothed aggregation AMG is carried out. The contour plots of the residual after five cycles of the smoothed aggregation AMG without and with local correction only on the finest grid are shown in fig. 6.2. Without any local correction, the large value of residual dominates in the region of low quality cells. With local correction on the finest grid, some large residuals at the centre are removed in the local correction area, but there are still parts of large residual persisting surrounding the low quality region, which is the high frequency error, as we predict, generated on the abstract coarse grids. The contour plot of the residual

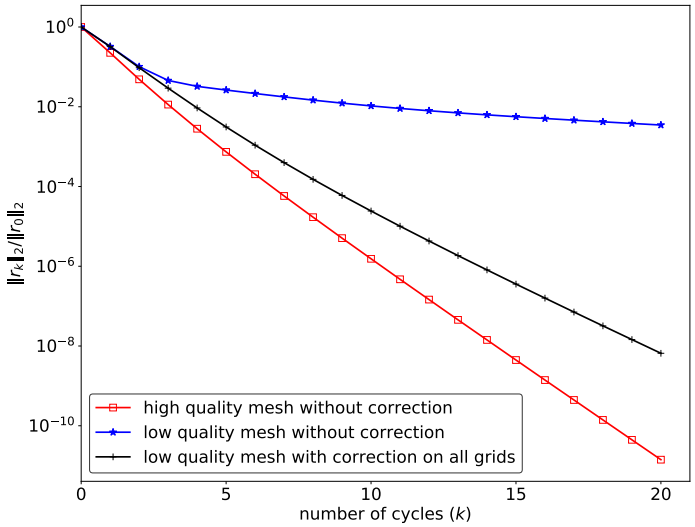


Fig. 6.3 Relative residual obtained by smoothed aggregation AMG on the low quality unit square mesh.

is different from which obtained by the classical AMG shown in fig. 5.3. It is observed that the residual is modest in other parts of the domain. Then the question becomes whether or not a reference level convergence rate can be achieved if we apply the local correction on the abstract coarse grids.

The relative residual with local correction on both fine and coarse grids after each smoothed aggregation AMG cycle is shown in fig. 6.3. It is found that even if applying the local correction on both fine and coarse grids, it is still unable to recover the convergence rate at the reference case of using the high quality mesh. Parts of residual are large in the regions of high quality cells, then the effect of the low quality cells becomes global. It is suggested that the main reason lies in the largest eigenvalue used in construing the smoothed prolongation operator.

6.1.2 Largest eigenvalue in smoothed prolongation

Recall that the smoothed prolongation operator of the smoothed aggregation AMG is defined in eq. (3.40). The smoothing procedure in constructing the prolongation allows the prolongation to connect neighbouring aggregates, and moreover, the smoothing step decreases the energy norm (eq. (3.36)) of the coarse grid basis functions. It is noted that the smoothing procedure is the weighted Jacobi which requires the value of the spectral radius of the system, i.e., $\rho(D^{-1}A)$. If a low quality mesh is used, then the maximum eigenvalue of the system is bounded above by the locally high frequency modes due to the low quality cells. The

smoothed prolongation generated by using the spectral radius of the whole system is then inefficient in the high quality regions.

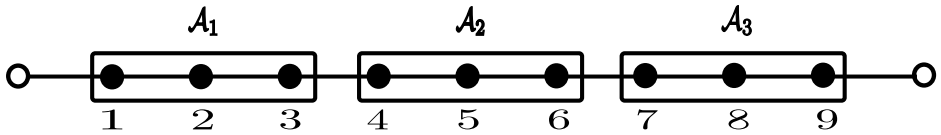
We consider an example of a one-dimensional line segment to illustrate what happens if the increased largest eigenvalue is used in the smoothed prolongation. The homogeneous Poisson equation is applied on the unit interval with nine interior nodes, as shown in fig. 6.4a. Suppose there are three aggregates around nodes 2, 5, and 8. Each aggregate covers three nodes. The spectral radius for the weighted Jacobi system is around 1.95. The coarse grid basis functions are constructed by different prolongation operators. Three plots of the coarse grid basis functions are shown, in which fig. 6.4b is obtained via the non-smoothed prolongation, fig. 6.4c is obtained using the smoothed prolongation with true largest eigenvalue. And, we intentionally increase the value of the radius ratio to 4, the coarse grid basis functions are shown in fig. 6.4d by using the smoothed prolongation with the increasing largest eigenvalue. The energy norm of the coarse grid basis functions in eq. (3.36) is also calculated for each case.

Clearly, the non-smoothed prolongation generates bad-looking basis functions on the coarse grid with a small connection between aggregates and high energy. Smoothed prolongation improves the connection between aggregates. If using the smoothed prolongation with the exact eigenvalue, the obtained coarse grid basis functions are exactly the ones for the P_1 finite element on the coarse grid. The energy norm of the coarse grid basis functions is minimised. If a large value for the spectral radius is used, the slope of the coarse grid basis functions becomes discontinuous, which results in more strain of the basis functions. It then leads to a raise in the energy norm of the basis functions on the coarse grid.

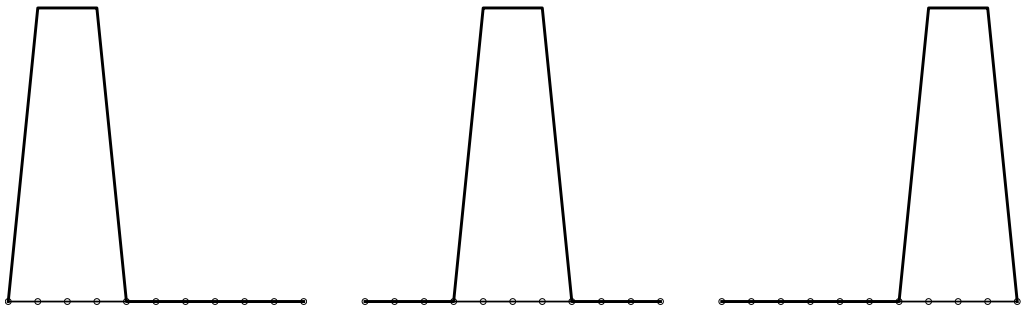
In the case of a mesh with a small number of locally poor quality cells, the system is well-conditioned in the high quality region, and the spectral radius of the whole system only 'suits' the local system in the low quality region. The energy norm of coarse grid basis functions generated by using the spectral radius of the whole system would be large. Due to the poor basis functions applied on the abstract coarse grids, then smoothed aggregation AMG becomes ineffective in high quality regions. And this is why even when applying the local correction on the coarse grids, we are still unable to obtain the convergence rate at reference level.

6.2 Shifted largest eigenvalue

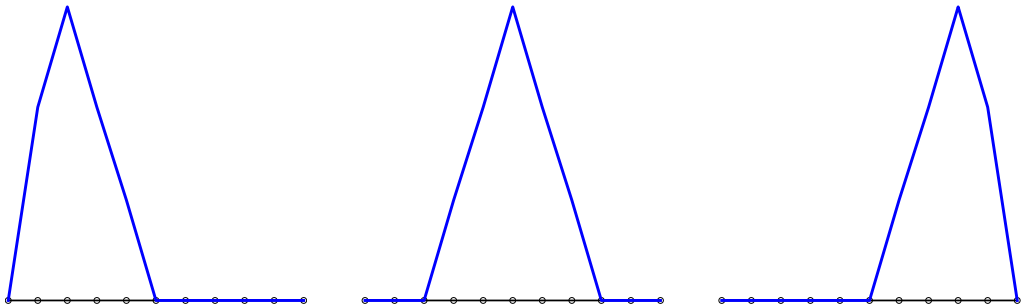
We propose a shifted largest eigenvalue strategy to shift the radius ratio to a reasonable value, which is similar to the corrected largest eigenvalue for the Chebyshev smoother as discussed



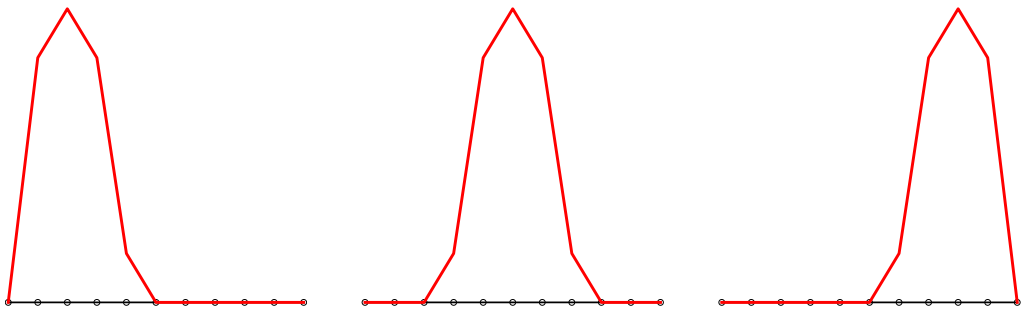
(a) Mesh and aggregates of the 1D line segment.



(b) Using non-smoothed prolongation
energy norm of the coarse grid basis functions = 20.



(c) Using smoothed prolongation with true spectral radius
energy norm of the coarse grid basis functions = 6.67.



(d) Using smoothed prolongation with increasing spectral radius
energy norm of the coarse grid basis functions = 10.

Fig. 6.4 The coarse grid basis functions obtained by different prolongation operators.

in section 4.3.4. The system is decomposed into four blocks

$$A = \begin{bmatrix} A_{gg} & A_{gb} \\ A_{bg} & A_{bb} \end{bmatrix}, \tag{6.1}$$

where A_{gg} corresponds to the system in high quality region, and A_{bb} corresponds to the system in low quality region where we apply the local correction. These block matrices can also be obtained algebraically. Let the set of indices corresponding to the DOFs in the high quality region $\Omega \setminus \Omega_B$ and the low quality region Ω_B be G and B, respectively. Let the operators $I_g : G \to N$ and $I_b : B \to N$ be the natural injections mapping from high quality region and low quality region to the whole domain. The four parts of the system can then be generated via the projection

$$\begin{aligned} A_{gg} &= I_g^T A I_g, \quad A_{gb} = I_g^T A I_b, \\ A_{bg} &= I_b^T A I_g, \quad A_{bb} = I_b^T A I_b. \end{aligned} \tag{6.2}$$

Our goal is to track the largest eigenvalue for the high quality region. Thus, the shifted largest eigenvalue with respect to the high quality region is taken as

$$\lambda_s = \rho(D_{gg}^{-1} A_{gg}). \tag{6.3}$$

Inserting the shifted eigenvalue, the smoothed prolongation operator is then written by

$$P_l = S_l \tilde{P}_l = \left(I - \frac{4}{3\lambda_s} D_l^{-1} A_l \right) \tilde{P}_l = \left(I - \frac{4}{3\rho({D_{gg}^l}^{-1} A_{gg}^l)} D_l^{-1} A_l \right) \tilde{P}_l, \tag{6.4}$$

where A_{gg}^l is the system on level l with respect to the high quality region, and D_{gg}^l is the diagonal matrix of A_{gg}^l. We emphasise here that this approximation is valid if the low quality region is much smaller than the high quality region. If the low quality region is very small, then the effects of A_{gb} and A_{bg} are small, and the shifted eigenvalue gives a more accurate spectral radius for high quality region.

6.3 Numerical results

Several numerical examples are presented in this section to validate the performance of the proposed local correction for smoothed aggregation AMG. Linear P_1 and quadratic P_2 bases are applied. The Ritz-Galerkin smoothed aggregation AMG (eq. (3.41)) is employed. Coarsening factors in smoothed aggregation AMG are chosen differently to fit different

Mesh level	1	2	3
High quality mesh	1.973297	1.505117	2.239345
Low quality mesh	2.968916	1.559350	2.240673
Shifted largest eigenvalue	1.973094	1.504936	2.240648

Table 6.1 Spectral radius and shifted largest eigenvalue used in the smoothed prolongation for the model problem.

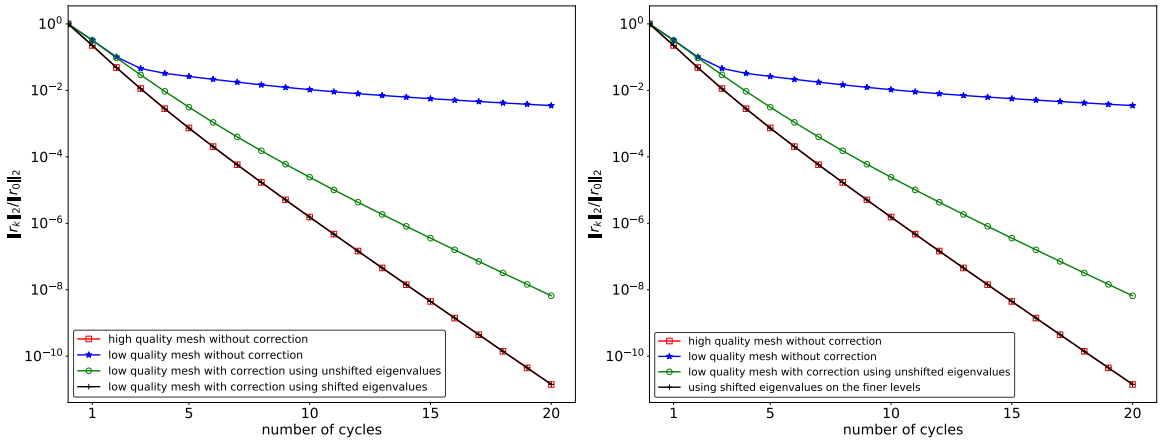
problem sizes. Symmetric Gauss–Seidel and Jacobi preconditioned Chebyshev methods are chosen as the global smoothers. The solver is terminated when the relative residual reaches 10^{-10}.

Unstructured meshes with simplex cells are created using the Gmsh [48]. Several nodes on the high quality mesh are perturbed to create the poor quality mesh. The low quality region Ω_B on the finest grid is tracked by the cells with $\gamma < 0.1$, where γ is the normalised radius ratio in eq. (4.1). Finite element tests are implemented by FEniCS [5, 80, 79], and the generation of aggregates and the transfer operators of the smoothed aggregation AMG is constructed using PETSc [11, 10, 12].

6.3.1 Model problem

The model problem discussed in section 6.1.1 is considered here. We have already seen that the performance of smoothed aggregation AMG is not satisfactory even if applying the local correction on all grids. Table 6.1 lists the spectral radius of the whole system and the shifted largest eigenvalue on each level. It is obvious that the spectral radius is large on the fine levels for the low quality mesh. On the other hand, it is seen that the increase in eigenvalues on the coarse levels is not that severe, and this is observed in all examples. It validates that the coarser levels in the smoothed aggregation AMG are relatively insensitive with respect to the low quality cells. The shifted largest eigenvalue strategy successfully captures the spectral radius for the high quality mesh. On the fine grid, the spectral radius of the high quality mesh and the shifted largest eigenvalue of the low quality mesh are almost identical since the low quality region is quite small. The convergence of the relative residual with local correction and shifted largest eigenvalue is shown in fig. 6.5a. The convergence rate of reference case is achieved with the shifted eigenvalue technique.

As discussed above, the sizes of the local correction systems on coarse grids are relatively large, which increases the computational costs on coarse grids. However, we have seen that the low quality cells have less influence on coarser levels in the smoothed aggregation AMG. Therefore, we only apply the local correction on the first two levels and make a comparison to the previous result. The computed convergence result with local correction only applied



(a) Local correction on all grids with shifted largest eigenvalues

(b) Local correction on first two levels with shifted largest eigenvalues

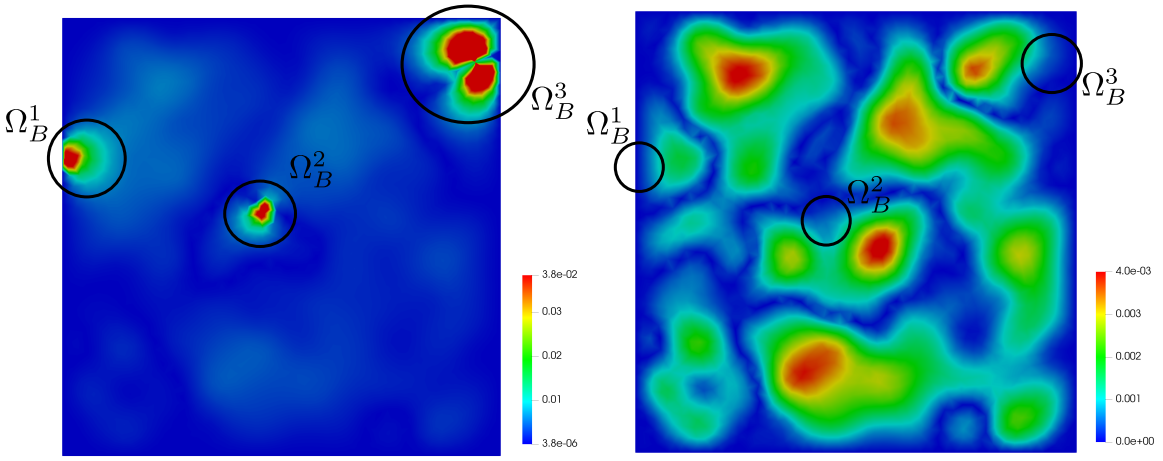
Fig. 6.5 Relative residual obtained by the smoothed aggregation AMG with local correction applied on all grids (a), and on first two levels (b) of the model problem.

on the first two levels is shown in fig. 6.5b. Remarkably, the convergence rate obtained by the local correction on the first two levels also matches the reference case, which supports our assertion that the low quality regions have only a minor effect on coarser grids.

6.3.2 Poisson equation on a unit square

We use again the unit square example shown in fig. 4.16, and solve the Poisson problem of eq. (5.4). Again the P_2 finite element is used for this small problem. The coarsening factor is chosen as 0.12 and the symmetric Gauss–Seidel is taken as the global smoother. Four–level smoothed aggregation AMG is carried out to solve this problem. The contour plots of the absolute value of residual obtained by the smoothed aggregation AMG without and with local correction only on the finest grid are shown in fig. 6.6. It is still obvious that, without local correction, the large value of residual localises in regions of low quality cells. However, with the local correction applied only on the finest grid, the pattern of large value residual is complicated. We can still see some large residual surrounding the low quality regions, especially the central one Ω_B^2. There are other large residuals also appearing in high quality regions, which is due to the use of the large spectral radius in the smoothed prolongation.

Table 6.2 lists the problem size of the finite element system and local correction system on each level. It can be seen that the coarsening is aggressive on the fine levels (especially the first two levels), then becomes less aggressive on the coarse levels. It results in a relatively large local correction system on level three, which takes up almost 25% of the whole problem on that level. Therefore, the local correction is omitted on the third level and applied only on



(a) Without local correction (b) With local correction on the finest grid

Fig. 6.6 Contour plots of the absolute value of residual on each vertex after ten cycles of the smoothed aggregation AMG on the unit square.

Mesh level	Number of DOFs in Ω	Number of DOFs in Ω_B
1	8633	292
2	1239	92
3	463	116
4	172	0

Table 6.2 Number of DOFs in the finite element and local correction system on each level of the unit square example for the smoothed aggregation AMG.

the first two levels. The spectral radius of the whole system and the shifted largest eigenvalue on each level are presented in table 6.3. The spectral radius is large for the low quality mesh on the fine levels, and the significant increase appears on the finest level. The shifted eigenvalue technique recaptures the 'corrected' largest eigenvalues in the high quality regions. On the other hand, it is surprising to see that the largest eigenvalue on the level three is even a little bit smaller for the low quality mesh, which provides the evidence that the local correction is not necessary on the coarser levels of the smoothed aggregation AMG case.

Mesh level	1	2	3
High quality mesh	1.973297	1.505117	2.239345
Low quality mesh	2.999283	1.618070	2.097266
Shifted largest eigenvalue	1.972952	1.488758	2.316939

Table 6.3 Spectral radius of the high and low quality meshes, and the shifted largest eigenvalues used in the smoothed prolongation for the unit square problem.

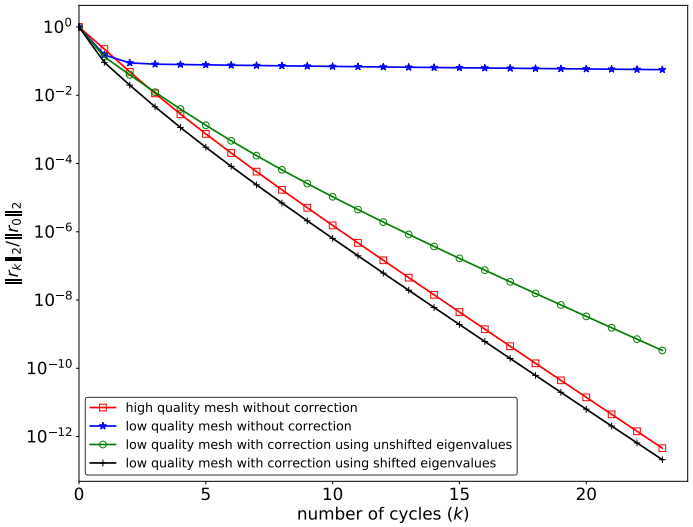


Fig. 6.7 Relative residual obtained by the smoothed aggregation AMG without local correction and with local correction on the first two levels.

FEM error	$\|u_{15} - u^*\|_{L_2(\Omega)}$
Element type	P2
High quality mesh	4.68×10^{-6}
Low quality mesh without correction	6.26×10^{-5}
Low quality mesh with correction	7.72×10^{-6}

Table 6.4 Finite element error in L_2 norm for the square problem, u_{15} is the approximated solution after for 15 multigrid cycles, and u^* is the exact solution to the PDE.

The convergence result obtained by the local corrections and the shifted largest eigenvalue is shown in fig. 6.7. The convergence rate with unshifted eigenvalues is still not fully recovered, though it may seem fast enough if we do not show the reference case. The shifted eigenvalue technique fully recovers the convergence rate at the reference case.

Again, we investigate the L_2 norm finite element error and list the result in table 6.4. The FEM error increases with low quality mesh. With the help of local correction, the error decreases to a satisfied level.

6.3.3 Linear elasticity on a lattice geometry

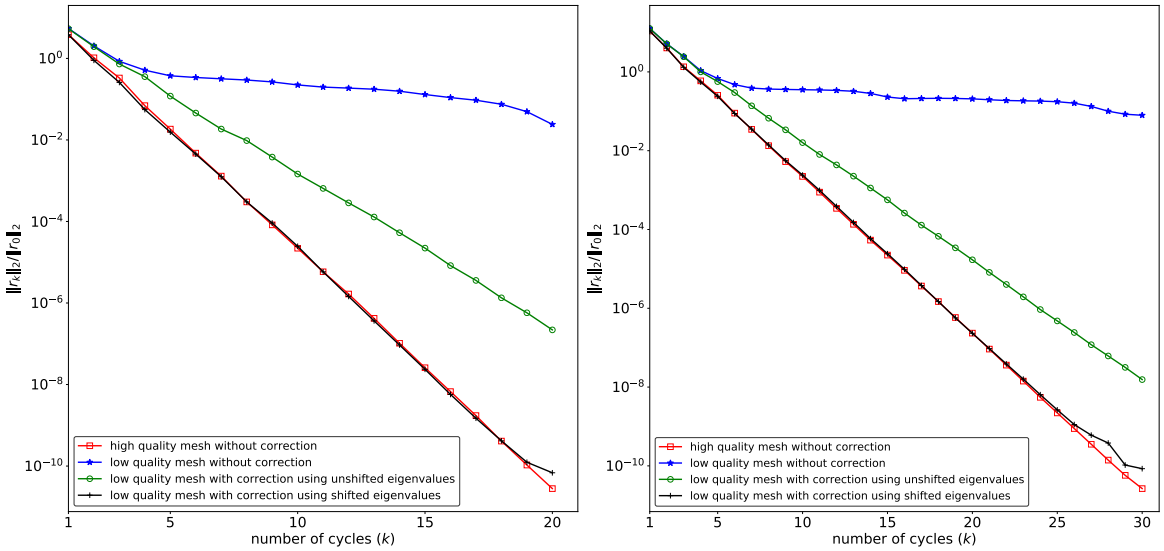
The strength of the smoothed aggregation AMG is its capability to solve vector-valued problems. We use the smoothed aggregation AMG preconditioned conjugate gradient method to solve the linear elasticity here, similar to the GMG case. It is emphasised that the

Element	Level	1	2	3	4
P_1 element	Spectral radius for high quality mesh	2.582721	1.901090	1.669230	2.659626
	Spectral radius for low quality mesh	6.993186	2.181098	1.633346	2.627887
	Shifted eigenvalue exact eigenvalue	2.582561	1.901110	1.687446	2.634350
P_2 element	Spectral radius for high quality mesh	3.634375	2.063461	1.607091	2.814008
	Spectral radius for low quality mesh	7.881907	2.244510	1.616285	2.634371
	Shifted eigenvalue exact eigenvalue	3.634468	2.063273	1.598310	2.737210

Table 6.5 Spectral radius for the high and low quality meshes, and shifted largest eigenvalues for the lattice problem.

setting of near null–space is essential in solving the vector-valued PDEs by the smoothed aggregation AMG, which is discussed in section 3.6.2. The lattice geometry shown in fig. 4.22 is used in this example. Linear elasticity is solved with fixed boundary conditions applied on both left- and right-hand sides of the domain, and a uniform force is applied downwards. The coarsening factor is chosen as 0.08 for P_1 element and 0.05 for P_2 element, which generates five–level mesh hierarchies. There are totally $437,946$ DOFs for P_1 element problem and $3,192,258$ DOFs for P_2 element problem. For this problem, we only apply the local correction and shifted eigenvalue strategies on the first two levels, and construct the smoothed prolongation with the spectral radius of the whole system on the other levels. The largest eigenvalues used for constructing the smoothed prolongation operators on different levels are listed in table 6.5. It is observed that the most noticed jump in the largest eigenvalues for the low quality mesh appears on the finest gird, the value increases by a factor of around two. On the coarse levels, the spectral radius for the high and the low quality meshes are at the similar numerical level. The shifted eigenvalues successfully recovers the 'corrected' eigenvalues in high quality regions on the fine grids.

The convergence results of the relative residual at each conjugate gradient iteration for both P_1 and P_2 elements are shown in fig. 6.8. Clearly, without using any local correction, the convergence of the smoothed aggregation AMG stagnates. The local correction with unshifted eigenvalues can improve the convergence, though it is still unable to match the speed of the reference level. The fully recovered convergence rate can be achieved by local correction with shifted eigenvalues on the fine levels. This example supports that even for a large 3D vector–valued problem, the convergence of the smoothed-aggregation AMG



(a) P_1 element (b) P_2 element

Fig. 6.8 Relative residual obtained by the smoothed aggregation AMG with and without local correction on the lattice problem.

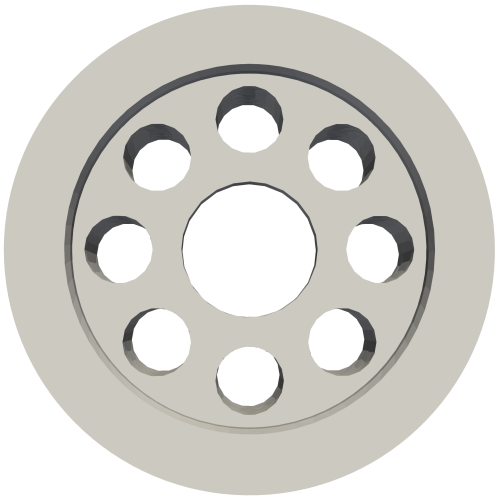
is relatively insensitive with respect to the coarser grids. Shifted eigenvalues and local correction used on the first several levels can be sufficient to fully recover the convergence rate. It is also noticed that the eigenvalue effects are stronger in P_1 element.

6.3.4 Linear elasticity on a pulley structure

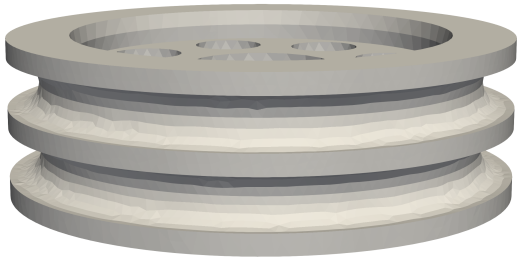
The example presented here is a pulley structure with several holes in the middle. The top view and side view of the geometry and the regions with low quality cells are shown in fig. 6.9. The linear elasticity equation is considered with homogeneous Dirichlet boundary conditions on the inner surface (holes) boundaries, and free surface on other parts of boundary. The loading (body) force models the centripetal acceleration with

$$f = (\rho\omega^2 x, \rho\omega^2 y, 0), \tag{6.5}$$

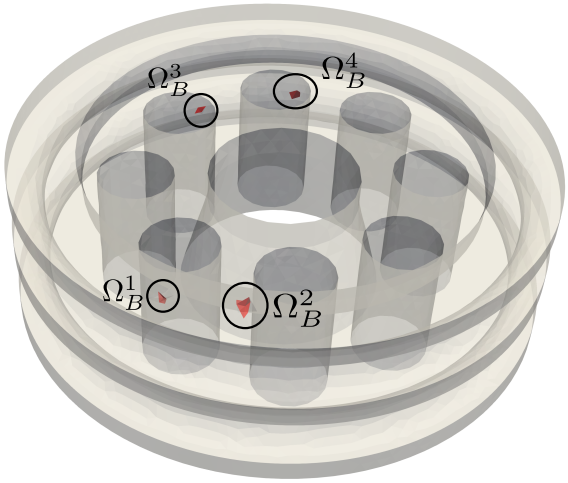
where $\rho = 10$ is the mass density and $\omega = 300$ is the rotation rate. The coarsening factor is still taken as 0.08 for P_1 element, and 0.05 for P_2 element. A five-level smoothed aggregation AMG is carried out for both elements. Symmetric Gauss–Seidel is taken as the global smoother. The local correction and shifted eigenvalue technique are applied on the first two levels, while the spectral radius of the whole system is used on other levels. The largest eigenvalues used to construct the smoothed prolongation operator on each level are reported



(a) Top view



(b) Side view



(c) Entire domain with low quality regions

Fig. 6.9 A pulley structure containing four regions of low quality cells.

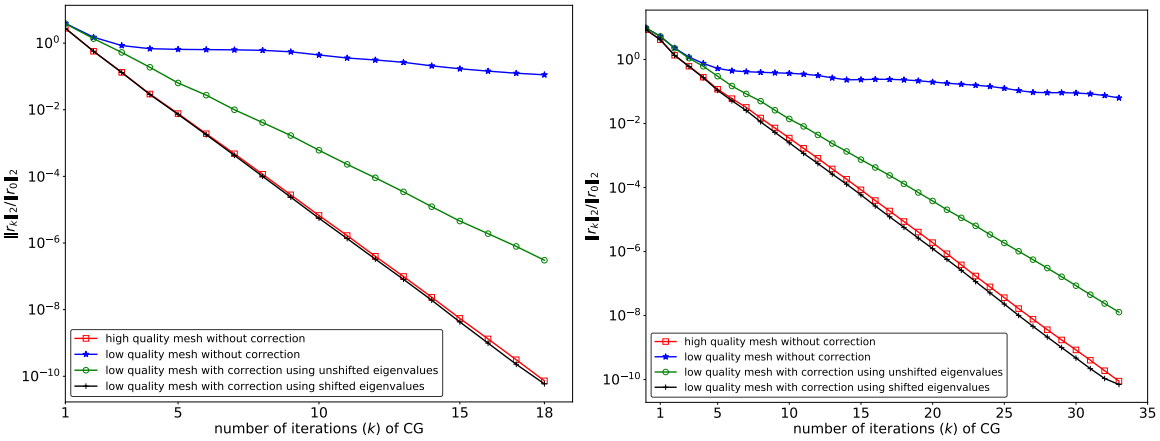
Element	Level	1	2	3	4
P_1 element	Spectral radius for high quality mesh	2.549472	1.890055	2.162992	2.337870
	Spectral radius for low quality mesh	7.160806	2.205719	1.818107	2.704671
	Shifted eigenvalue for low quality mesh	2.549419	1.890065	2.163016	2.645824
P_2 element	Spectral radius for high quality mesh	4.142785	2.048754	1.773213	2.304506
	Spectral radius for low quality mesh	8.027026	2.362346	1.861919	2.158552
	Shifted eigenvalue for low quality mesh	4.142518	2.049212	1.808892	2.202773

Table 6.6 Spectral radius and shifted largest eigenvalues used in constructing the smoothed prolongation for the pulley example.

in table 6.6. Similarly, the value of the spectral radius is quite large for the low quality mesh on the finer levels. The spectral radius for both high and low quality meshes on the coarse levels are similar in numerics. Shifted largest eigenvalues on the fine grids match perfectly with the spectral radius obtained by using the high quality mesh.

The relative residual after each iteration of the smoothed aggregation AMG preconditioned conjugate gradient method for both element types is shown in fig. 6.10. Local correction helps improve the convergence rate dramatically, especially for the P_2 element. With the sifted eigenvalues used in the smoothed prolongation, the convergence rate of the reference case is achieved.

Exact computational time for the pulley problem is listed here. We still split the whole procedure in terms of the three components: FEM set–up, multigrid set–up, and multigrid preconditioned CG iterations. The time costs are listed in table 6.7. For both element types, convergence completely stops for the low quality mesh. With local correction, the convergence is recovered in reasonable time. It takes $30-40\%$ more time for the CG solver iterations with local correction, which is similar to previous examples. However, it is noted that there is a significant increment in time for multigrid set–up procedure in the smoothed aggregation AMG case. The reason comes from the construction of the new transfer operators. The transfer operators are not changed in the case of GMG and classical AMG, while the local correction scheme for smoothed aggregation AMG requires to reconstruct new prolongation operators since we use the shifted largest eigenvalues. It is found that construction of the new transfer operators takes a large amount of time in our implementation. On the other

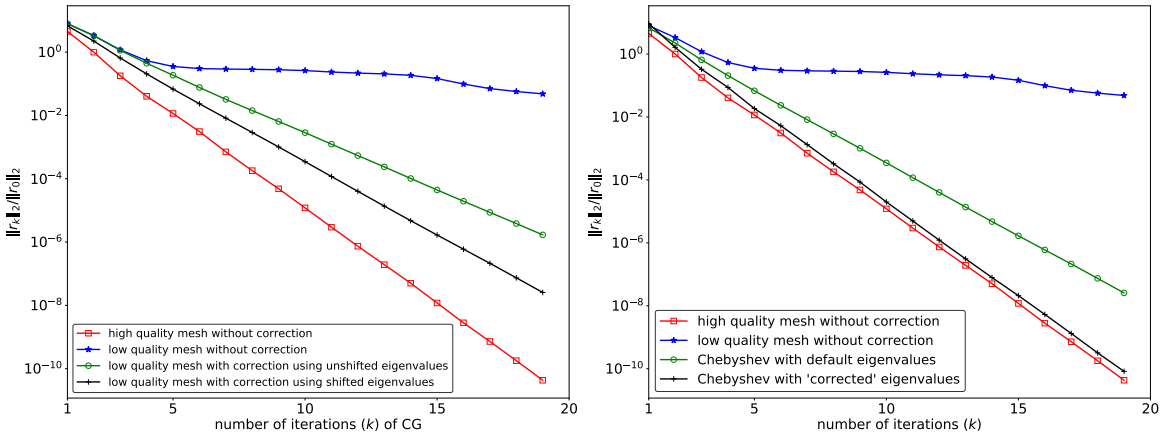


(a) P_1 element (b) P_2 element

Fig. 6.10 Relative residual obtained by smoothed aggregation AMG without and with local correction using the symmetric Gauss–Seidel smoother for the pulley problem.

Element type	Mesh quality	FEM set–up	Multigrid set-up	CG iterations
P_1	High quality mesh	7.25	73.84	67.41
	Low quality mesh	7.23	73.82	>3600
	Low quality mesh with correction	7.27	143.65	87.54
P_2	High quality mesh	52.77	512.31	706.01
	Low quality mesh	52.99	510.69	>3600
	Low quality mesh with correction	53.14	1102.43	984.42

Table 6.7 Computational time in seconds for relative residual to reach 10^{-10} solving the pulley problem using the smoothed aggregation AMG, in which there are three components: (i) FEM system construction, (ii) multigrid set–up including constructing coarse grid systems and transfer operators, tracking low quality cells, and making local correction systems, and (iii) multigrid preconditioned CG iterations.



(a) Chebyshev smoother with true largest eigenvalues

(b) Chebyshev smoother with corrected largest eigenvalues

Fig. 6.11 Relative residual obtained by smoothed aggregation with and without local correction using Chebyshev smoother on the pulley structure.

hand, this procedure can be implemented more efficiently via parallelism. We also make a proposed approach in section 9.2.3.

We now test this problem with the Jacobi preconditioned Chebyshev smoother which is commonly employed in the content of the smoothed aggregation AMG for engineering problems [1]. If applying the same procedure in the Gauss–Seidel case, the convergence result for the P_1 element problem is shown in fig. 6.11a. First, notice that the convergence also stagnates for the low quality mesh without using local correction. With the help of the local correction and shifted eigenvalue strategy, the convergence rate improves more. However, it is still unable to match the speed of the reference case. The cause is the use of the increasing maximum eigenvalues in constructing the Chebyshev smoother, as discussed in section 4.3.4. It is necessary to use the corrected largest eigenvalue in section 4.3.4 to construct the Chebyshev smoother. The relative residual obtained by using the corrected largest eigenvalues in the Chebyshev smoother is shown in fig. 6.11b. The convergence rate with the corrected largest eigenvalue in Chebyshev smoother matches the convergence rate of the reference case. This example validates the idea of shifted eigenvalue technique in the smoothed aggregation AMG and the Chebyshev smoother for locally poor quality complicated engineering meshes.

6.4 Concluding remarks

We discover that to fully recover the poor convergence rate of the smoothed aggregation AMG on low quality meshes, it is necessary to use a reasonable spectral radius to construct a proper smoothed prolongation operator. A shifted largest eigenvalue technique is proposed to shift the largest eigenvalue of the whole system to a 'corrected' value corresponding to the high quality region of the system. Numerical experiments validate that the shifted largest eigenvalue matches the spectral radius obtained by the high quality mesh. With the local correction and shifted largest eigenvalue applied on the first several fine levels of smoothed aggregation AMG, the poor convergence can be restored to the reference case. This algebraic AMG approach is particularly appealing at extreme scale computation, e.g., contact mechanics, which is well suitable for extreme parallelisation.

Chapter 7

Nonlinear problems

The second last step to reach our goal of solving the complicated engineering problem is the extension of the local correction method to solve nonlinear problems. Nonlinear problems are largely encountered in engineering applications, and low quality meshes are not uncommon. When using the Newton's method to solve large size nonlinear problems, it is necessary to employ iterative solvers to solve the corresponding Jacobian system, the speed of which is still worsened by poor quality meshes. Therefore, the proposed multigrid with local correction scheme can be perfectly embedded in the Newton's method as the inner solver to deal with problems containing low quality cells. Then, the main problem becomes whether or not the inner multigrid solver with local correction is sufficient, and whether or not we need to make changes to the outer Newton's solver.

This chapter aims to validate whether the Newton–multigrid with local correction can recover the convergence rate on low quality meshes to the reference level via numerical examples. The local correction used in GMG, classical AMG and smoothed aggregation AMG are all tested as parts of inner solvers for the Newton's method to solve different types of nonlinear problems. In section 7.1 and section 7.2, the Newton's method and the Newton–multigrid solver are explained. A model problem of the nonlinear Poisson equation on a low quality unit square mesh is illustrated in section 7.3 to show that the multigrid with local correction can be served as a satisfactory inner solver to improve the convergence rate of the Newton's method on low quality meshes to the reference level as desired. Numerical examples are carried out in section 7.4, and we intentionally choose two complicated nonlinear equations, including a p-Laplacian problem in section 7.4.1 and a hyperelasticity problem in section 7.4.2. Concluding remarks are presented in section 7.5.

7.1 Newton's method

If we solve a nonlinear elliptic type PDE, then a nonlinear weak form is obtained, which is generally written as

$$W(u,v) = 0, \quad \forall v \in V, \tag{7.1}$$

where $W(u,v)$ is an operator which is nonlinear in the solution u, and linear in the test function v. After the finite element discretisation, a nonlinear system with n nonlinear algebraic equations containing n unknowns

$$F(u) = 0, \tag{7.2}$$

is then generated where F is a nonlinear operator. The Newton's method [32, 66] is an efficient method for solving nonlinear algebraic systems generated from nonlinear PDEs. It is based on the linearisation of nonlinear equations. Expand the nonlinear system eq. (7.2) with a perturbation du by a Taylor series

$$F(u+du) = F(u) + Jdu + O(du^2), \tag{7.3}$$

where J is the Jacobian matrix of the nonlinear operator with $[J]_{ij} = \partial F_i / \partial u_j$, i.e.

$$J = \begin{bmatrix} \frac{\partial F_1}{\partial u_1} & \frac{\partial F_1}{\partial u_2} & \cdots & \frac{\partial F_1}{\partial u_n} \\ \frac{\partial F_2}{\partial u_1} & \frac{\partial F_2}{\partial u_2} & \cdots & \frac{\partial F_2}{\partial u_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial F_n}{\partial u_1} & \frac{\partial F_n}{\partial u_2} & \cdots & \frac{\partial F_n}{\partial u_n} \end{bmatrix}. \tag{7.4}$$

The higher order terms in the Taylor series eq. (7.3) are omitted, then a linear system of the perturbation du is generated

$$Jdu = -F(u). \tag{7.5}$$

Thus, the Newton's method updates the solution by finding the perturbation in each iteration via

$$u_{k+1} = u_k - J^{-1}(u_k)F(u_k). \tag{7.6}$$

The beauty of the Newton's method is transforming the nonlinear problem to a sequence of linear problems. However, the time costly procedure in Newton's method is still often the inversion of the Jacobian matrix. For a large problem, iterative solvers are also employed to solve the Jacobian matrix in the Newton's method. These methods are called Newton-inexact methods, and if a Krylov-type iterative solver is applied, it is then called a Newton–Krylov

solver. The convergence of Newton–inexact method is highly dependent on the performance of the inner linear solver [31, 66], and different linear inner solvers may suit different types of nonlinear problems [37, 99]. Another main feature of Newton's method is its sensitivity with respect to the initial guess. Without a good initial guess, it is highly possible that the method diverges or even converges to some other solution. In order to solve the nonlinear PDEs, an (approximated) solution to a linear variance of the nonlinear problem is usually taken as the initial guess for Newton's method.

7.2 Newton-multigrid method

A natural way to employ multigrid method to solve nonlinear problems is to embed it in the Newton solver to solve the Jacobian system, which results in the Newton–multigrid method. The convergence analysis and error bounds for Newton–multigrid method has been well studied in [108]. We make an important remark here that there is another way to exploit multigrid to solve the nonlinear system, which is the full approximation scheme. Brandt [24] proposes the full approximation scheme (FAS) which is a nonlinear variance of the multigrid V-cycle. It is found that FAS can solve nonlinear Poisson type equations with similar convergence rate to the linear case [92]. Several comparisons between Newton–multigrid and FAS have been carried out, e.g., [53, 18]. Many have observed that the performance of FAS is weaker than Newton-multigrid, especially for complicated engineering problems.

It has been already shown that the convergence of multigrid degrades dramatically when encountering low quality cells. If Newton-multigrid method is employed to solve a nonlinear problem on a locally poor quality mesh, then the step of inversion of the Jacobian matrix is not feasible due to the failure of multigrid. As a result, Newton's method also fails automatically with the inaccurate and non-improving solution to the the Jacobian system. Then we will ask the question: can the proposed local correction used in multigrid improve the convergence of Newton-multigrid method, and is it also necessary to make changes to Newton's solver?

7.3 A model problem

A simple nonlinear Poisson equation on a unit square $\Omega = (0,1)^2$ is set as the model problem with

$$\begin{aligned} -\nabla \cdot (q(u)\nabla u) &= \sin(\pi x)\sin(\pi y) \quad \text{in } \Omega, \\ q(u) &= 1+u, \\ u &= 0 \quad \text{on } \partial\Omega \end{aligned} \tag{7.7}$$

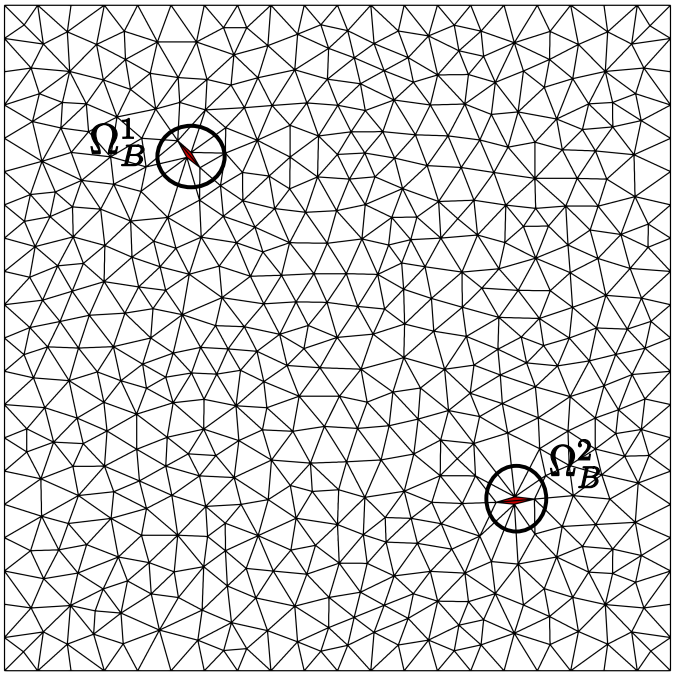
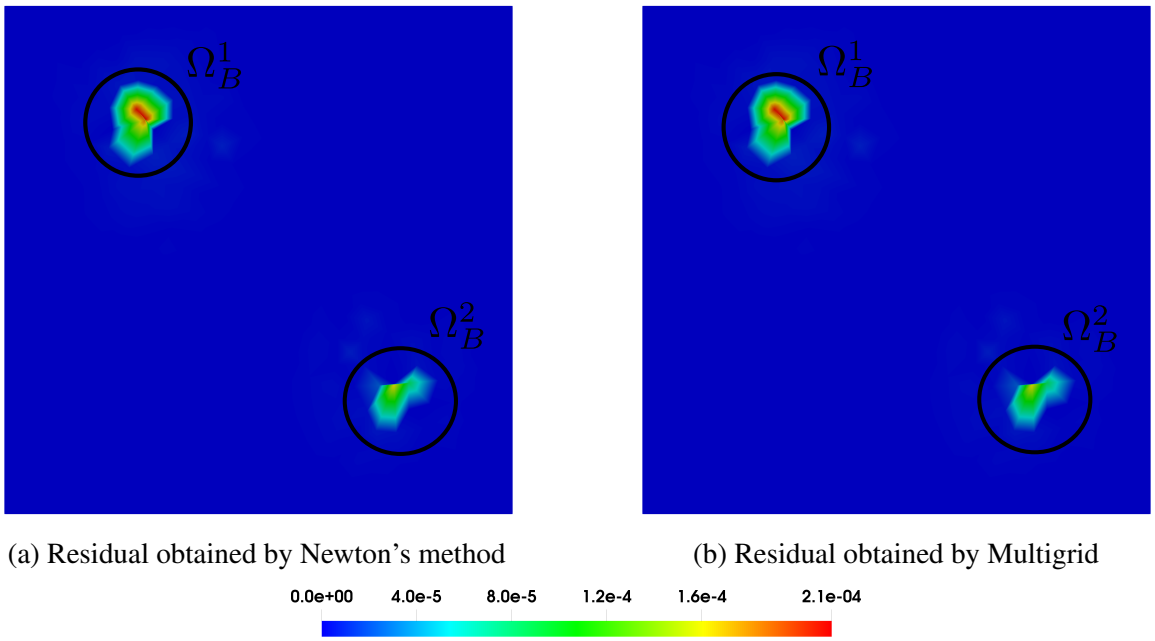


Fig. 7.1 A unit square mesh with two locally poor quality regions.

A low quality unit square mesh with two locally poor quality regions is used and shown in fig. 7.1. The mesh contains 148 nodes and 253 cells. P_1 finite element method is carried out. We first test the Newton–multigrid method with a two-level geometric multigrid served as the inner solver. The coarse grid of the GMG is kept of high quality. Symmetric Gauss–Seidel is taken as the smoother, and one iteration of the smoother is applied in terms of pre- and post-smoothing. For each iteration of Newton's method, five applications of multigrid V-cycle is used to solve the Jacobian system. Initial guess for the Newton's method is taken by the solution to the linear Poisson's equation approximated by the linear multigrid V-cycle. The absolute value of residual on each vertex obtained after three iterations of Newton's method as well as the residual obtained by the inner multigrid V-cycle at the last Newton iteration are shown in fig. 7.2. It is not surprising to see that the only remaining large value residual collapse the regions of low quality cells. Since multigrid can not remove the high frequency error, then the perturbation term in Newton's method is always inaccurate in regions of low quality cells. On the other hand, it is noted that the residual remained in both the Newton's method and the inner multigrid cycles are kept at the same value. This validates that the degraded performance of Newton–multigrid comes directly from the poor performance of the inner multigrid solver in solving the Jacobian matrix. Moreover, if these large value residual produced by multigrid can be resolved by the local correction, then it is anticipated that the convergence of Newton's method can be totally recovered.

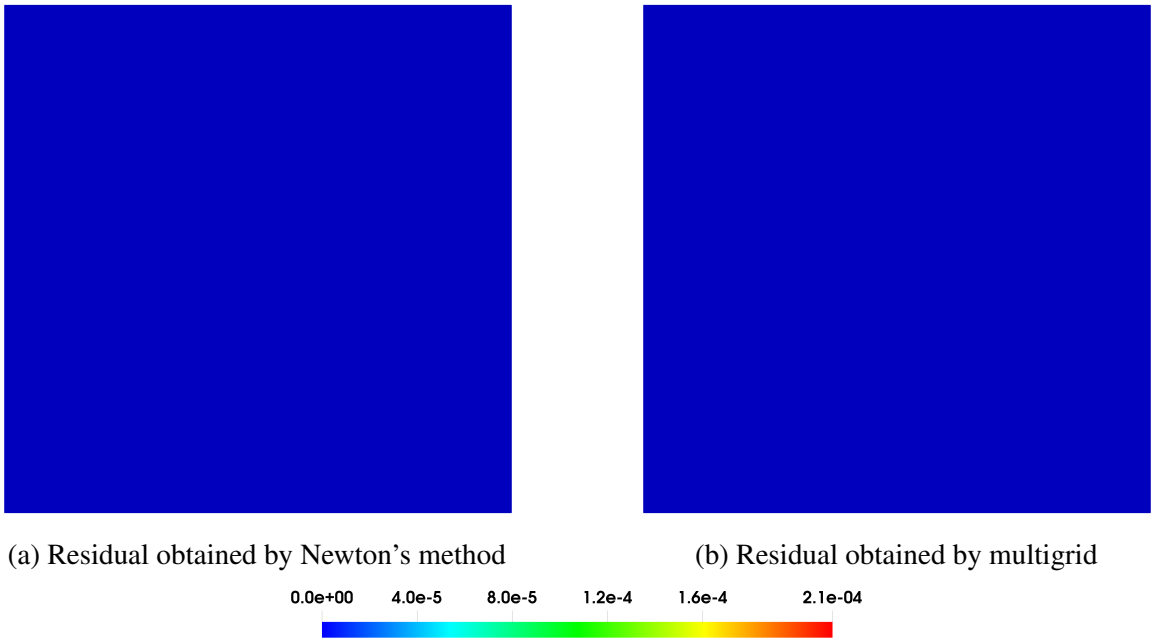


(a) Residual obtained by Newton's method (b) Residual obtained by Multigrid

Fig. 7.2 Absolute value of residual after three iterations of Newton-multigrid methods (a) and absolute value of residual obtained by multigrid inner solver at the last Newton iteration (b).

After applying multigrid with local correction as the inner solver, the absolute value of residual on each vertex after three iterations of the Newton method as well as the residual obtained by inner multigrid solver at the last Newton iteration are shown in fig. 7.3. The local correction removes the large value of residual in low quality regions produced by inner multigrid solver. The perturbation term of the Newton's method is then 'corrected', and the Newton's method can be fully recovered. The relative residual versus iteration of Newton's method with the inner solvers chosen by direct method and multigrid is shown in fig. 7.4. If a direct solver is served as the inner solver, Newton's method converges quadratically. It is found that the cell quality has a small effect on convergence of the Newton exact method. With the poor quality mesh, the convergence of the Newton–multigrid method becomes dramatically slow after the third iteration. A full recovery of convergence rate of the Newton–multigrid method at reference case is achieved if we apply the local correction for the inner multigrid solver. The examples validates that the dominant residual obtained by the Newton–multigrid solver comes from the inner multigrid solving the linear Jacobian equation. With the local correction used in inner multigrid solver, the convergence improves significantly for both outer and inner methods, and reaches the level of using the high quality mesh, which is desired.

Classical AMG is also effective in solving scalar–valued Poisson type problem, and tested for the model problem. Three–level mesh hierarchies are applied. For each iteration



(a) Residual obtained by Newton's method (b) Residual obtained by multigrid

Fig. 7.3 Absolute value of residual after three iterations of Newton-multigrid methods with local correction (a), and absolute value of residual obtained by inner multigrid solver at last Newton iteration (b).

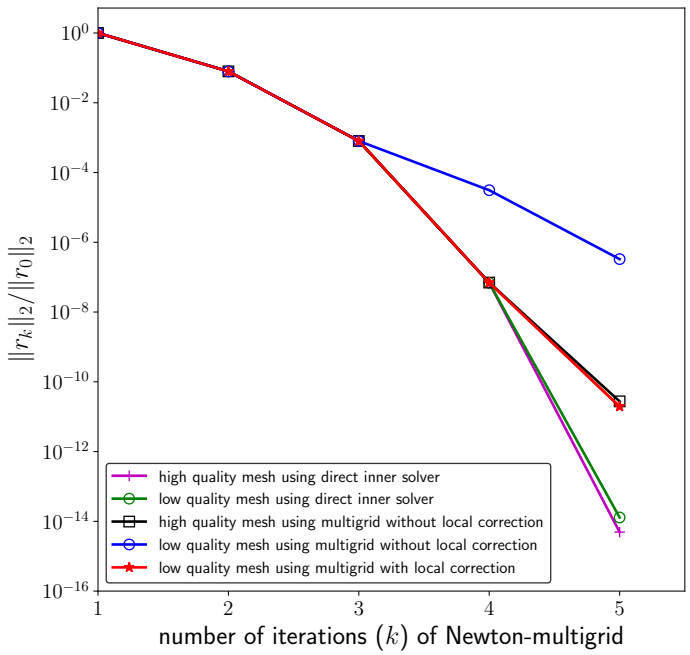
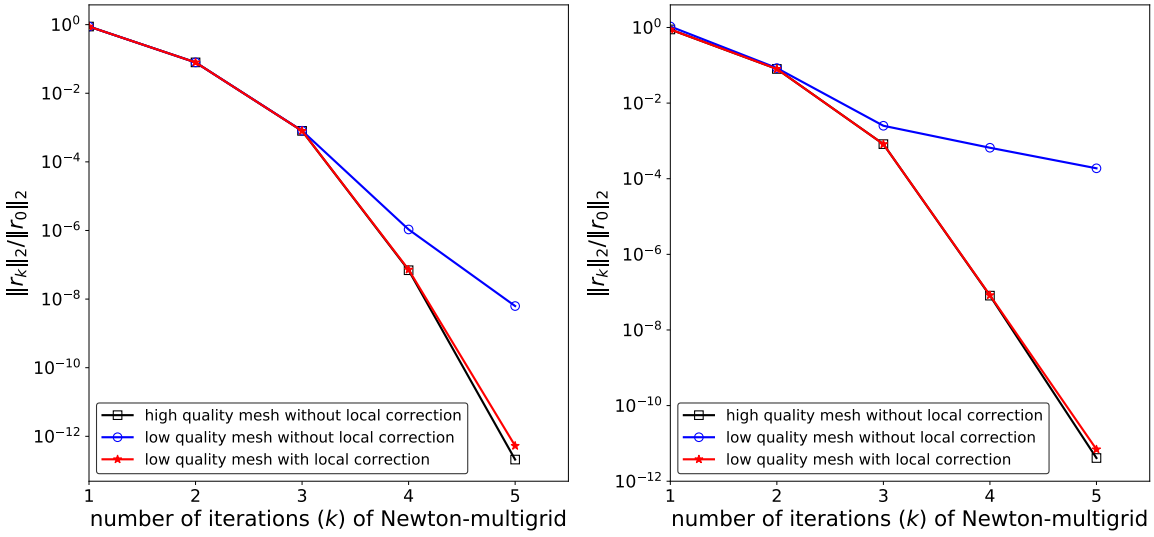


Fig. 7.4 Relative residual of Newton's method with inner solvers chosen by direct method and multigrid for the nonlinear Poisson problem.



(a) P_1 element

(b) P_2 element

Fig. 7.5 Relative residual of the Newton-multigrid method with classical AMG used as the inner solver working on the nonlinear Poisson problem.

of Newton's method, five cycles of classical AMG are carried out as the inner solver. Symmetric Gauss–Seidel is still taken as the global smoother, and applied twice for pre- and post-smoothing. The local correction scheme used for the classical AMG is described in section 5.2. The relative residual of the Newton's method with classical AMG as the inner solver is shown in fig. 7.5 for both P_1 and P_2 elements. It is noted that if AMG is served as the inner linear solver, the computational cost of the Newton-multigrid method is higher than using GMG as the inner linear solver. For the GMG case, the transfer operators, P and R, are based on the geometries of mesh hierarchies and will not change through Newton iterations since mesh levels are fixed. However, the operators P and R of AMG are based on the algebraic information of the target matrix, which is the Jacobian matrix in the Newton's method. The Jacobian matrix is not fixed during each iteration of the Newton's method, so that we need to generate the new P and R for AMG at each iteration of the Newton method, which takes more computational cost in Newton-AMG case. Without local correction, the Newton-AMG method also converges much slower from the third iteration. The convergence rate of the Newton's method can be fully recovered if the local correction scheme proposed in section 5.2 is applied for the inner classical AMG. The method works well for both P_1 and P_2 elements of the nonlinear PDEs, which is promising in solving large complex engineering applications.

7.4 Numerical results

The performance of the Newton-multigrid with local correction is examined here by numerical examples. The construction of multigrid is the same in previous numerical examples, i.e. GMG in section 4.5, classical AMG in section 5.3, and smoothed aggregation AMG in section 6.3. Symmetric Gauss–Seidel is taken as the global smoother. Two applications of smoothers are applied in pre- and post-smoothing of multigrid. Initial guess for the Newton's method is obtained via solving the linear variance of the nonlinear problem. Local correction schemes for GMG (chapter 4), classical AMG (chapter 5), and smoothed aggregation AMG (chapter 6) are all tested.

Meshes are unstructured with simplex cells, which are created by the Gmsh [48]. The low quality mesh is generated by moving a number of vertices in a high quality mesh. The low quality regions Ω_B is still determined by tracking the cells with $\gamma < 0.1$, where γ is the normalised radius ratio in eq. (4.1). The nonlinear finite element is constructed by FEniCS [5, 80, 79] as well as the corresponding Jacobian matrix. The generic Newton-multigrid is implemented using PETSc [11, 10, 12], which is the same in previous examples.

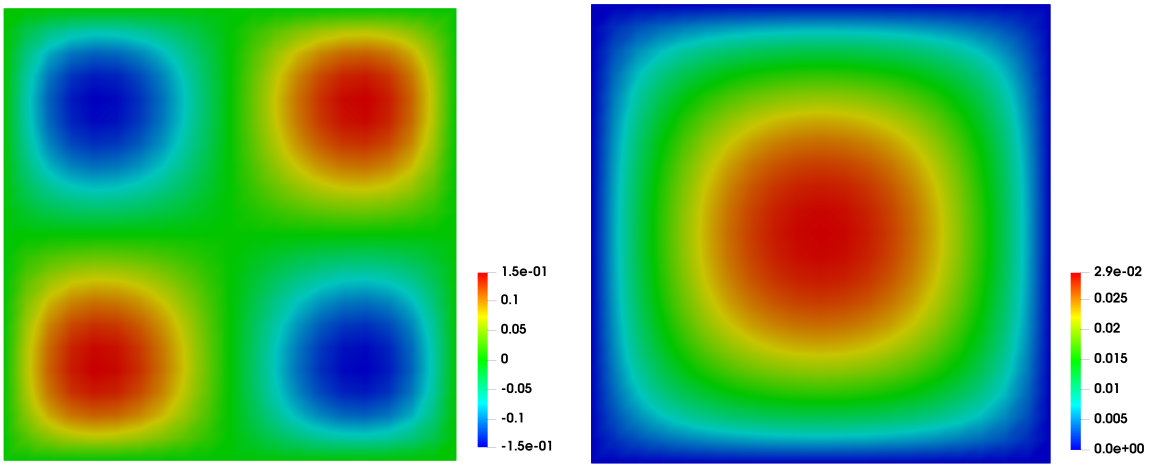
7.4.1 p-Laplacian on a unit square

The regularised p-Laplacian formulation [28] is tested here which is also in the form of the nonlinear Poisson type equation with

$$\begin{aligned} -\nabla \cdot (q(u)\nabla u) &= f \quad \text{in } \Omega, \\ q(u) &= \left(\varepsilon^2 + \frac{1}{2}|\nabla u|^2\right)^{(p-2)/2}, \end{aligned} \tag{7.8}$$

where ε is the regularization parameter and p refers to the exponent of the Laplacian. Notice that when $p = 2$ and $\varepsilon = 0$, the p-Laplacian reduces to a linear Poisson equation. We consider the case in which $p = 5$, and the right hand side $f = 0.1$, which makes the problem into the nonlinear regime. The regularization parameter is set as $\varepsilon = 10^{-2}$, which makes the problem anisotropic. The domain tested is the unit square mesh shown in fig. 4.16 and the initial guess is taken by $u_0(x,y) = xy(1-x^2)(1-y^2)$. The initial guess and the converged solution for high quality mesh are shown in fig. 7.6. The final solution is localised at the middle of the square, so that the problem has difficult local nonlinearity that global solvers may not be able to solve.

Newton–multigrid is carried out with the inner solver taken by the classical AMG and smoothed aggregation AMG since AMG is good at solving the anisotropic problems. Five



(a) Initial guess u_0 (b) Converged solution

Fig. 7.6 Initial guess u_0 and the converged solution of the 5-Laplacian problem obtained by a direct solver.

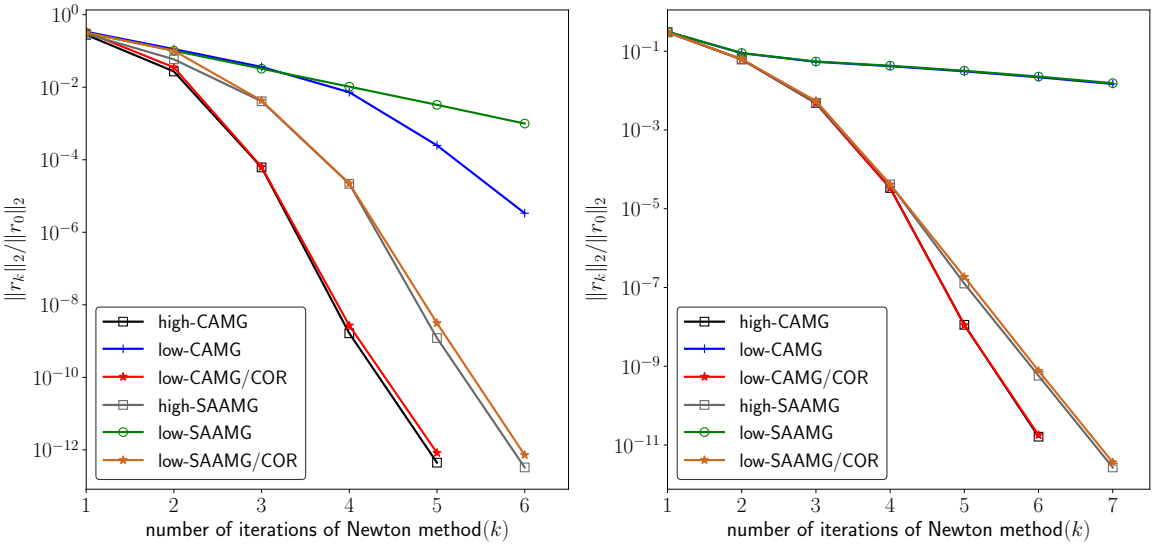
multigrid V-cycles are applied as the inner solver in each iteration of Newton's method. Symmetric Gauss–Seidel is taken as the global smoother. The local correction on all levels (except the coarsest one) including the abstract coarse grids introduced in section 5.2 and the shifted largest eigenvalue technique for the smoothed aggregation AMG described in section 6.2 are employed. The relative residual of the Newton-AMG methods with iteration count is shown in fig. 7.7 for P_1 and P_2 elements.

Without local correction, the Newton-AMG method for both elements stagnates even for this simple mesh. The local correction schemes can recover the slow convergence to the reference level for both the classical and smoothed aggregation AMG. The example shows that the local correction in the inner multigrid solver is sufficient to recover the convergence of the Newton's method, especially for this anisotropic problems. An impressive discovery is that the convergence rate of using the classical AMG is a little bit faster than using the smoothed aggregation AMG in this example. It supports that classical AMG fits well the scalar-valued problems, especially the Poisson type PDEs.

7.4.2 Hyperelasticity on a unit cube

Let the domain of the problem be the unit cube $\Omega = (0,1)^3$, as shown in fig. 5.8. Hyperelasticity is based on minimising the total potential energy Π over the displacement field u with

$$\min_u \Pi(u), \quad \Pi(u) = \int_\Omega \psi(u)dx - \int_\Omega B \cdot u\,dx - \int_{\partial\Omega} T \cdot u\,ds \tag{7.9}$$



(a) P_1 element (b) P_2 element

Fig. 7.7 Relative residual of the Newton-multigrid method with AMG as the inner solver working on the 5-Laplacian problem, CAMG stands for classical AMG, SAAMG means the smoothed aggregation AMG, and COR refers to the use of local correction.

where B is a uniform body force, T is a traction force, and ψ is the elastic stored energy density. We consider the common neo-Hooken stored energy model with the energy density

$$\psi = \frac{\mu}{2}(I_c - 3) - \mu \ln(J) + \frac{\lambda}{2}\ln(J)^2, \tag{7.10}$$

in which μ and λ are Lamé constants, Ic and J are some scalars defined by

$$\begin{aligned} J &= \det(F), \\ I_c &= \mathrm{trace}(C), \end{aligned} \tag{7.11}$$

where F is the deformation gradient

$$F = I + \nabla u, \tag{7.12}$$

and C is the right Cauchy Green tensor

$$C = F^T F. \tag{7.13}$$

A soft material property is chosen with the Young's modulus $E = 10.0$, Poisson ratio $\nu = 0.3$ and the unit volume body force $B = (0, -0.5, 0)$. Several boundaries are defined for the unit

cube geometry with

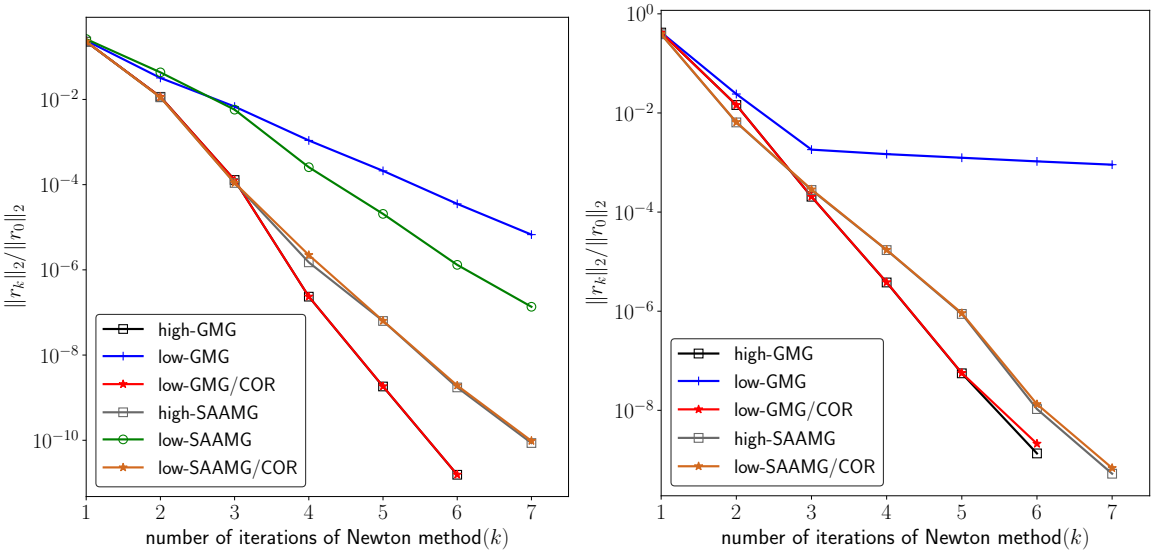
$$\begin{aligned}\Gamma_D^1 &= 0 \times (0,1) \times (0,1),\\ \Gamma_D^2 &= 1 \times (0,1) \times (0,1),\\ \Gamma_N &= \partial\Gamma \setminus (\Gamma_D^1 \cup \Gamma_D^2).\end{aligned} \tag{7.14}$$

The mixed boundary conditions are considered, imposed via

$$\begin{aligned}u =& (0,0,0) \quad \text{on } \Gamma_D^1\\ u =& (0,\\ & (0.5 + (y-0.5)\cos\theta - (z-0.5)\sin\theta - y)/2,\\ & (0.5 + (y-0.5)\sin\theta - (z-0.5)\cos\theta - z)/2) \quad \text{on } \Gamma_D^2\\ T =& (1,0,0) \quad \text{on } \Gamma_N.\end{aligned} \tag{7.15}$$

Newton-multigrid is carried out and the inner solver is chosen by the multigrid preconditioned conjugate gradient method. Both geometric multigrid and smoothed aggregation AMG are served as the preconditioner for the conjugate gradient method. Four level mesh hierarchies are used for both multigrid methods. For each iteration of Newton method, two iterations of multigrid preconditioned conjugated gradient is applied as the inner solver, in which two multigrid V-cycles are used as the preconditioner. Symmetric Gauss–Seidel is chosen as the global smoother. The initial guess is taken by the solution of the linear elasticity obtained by ten cycles of linear multigrid.

The relative residual of the Newton's method with and without local correction is shown in fig. 7.8 for P_1 and P_2 elements. It is noted that for the P_2 element, the Newton–multigrid method diverges with the low quality mesh, which is due to the inaccurate initial guess. Clearly, without local correction, Newton–multigrid fails for using the low quality mesh, especially for P_2 element. The local correction schemes improve the performance of multigrid preconditioned conjugate gradient, and recovers the convergence rate of the Newton method to the reference case. It is also observed that convergence rate of using GMG preconditioner as the inner solver is better than using the smoothed aggregation AMG. Full geometric mesh hierarchies are required for GMG to construct the transfer operators, which leads to a better convergence than AMG. The example validates the capability of the proposed local correction smoother used in the Newton–multigrid solver for large–size complicated vector–valued problems.



(a) P_1 element (b) P_2 element

Fig. 7.8 Relative residual obtained by Newton-multigrid method with multigrid preconditioned conjugate gradient method as the inner solver working on the hyperelasticity problem, GMG stands for geometric multigrid, SAAMG means the smoothed aggregation AMG, and COR refers to the use of local correction.

7.5 Concluding remarks

In this chapter, it is found that convergence of Newton–multigrid method still degrades dramatically with low quality meshes. The reason behind it is the inaccurate solution to the Jacobian system solved by the inner multigrid method due to the low quality cells. With the help of local correction, the poor performance of Newton–multigrid method can be fully recovered. An impressive finding is that local correction for inner multigrid solver is adequate to improve the poor convergence rate of Newton–multigrid method to the reference level, and there is no need to make other treatments for Newton's method. The algorithm has been tested for both GMG and AMG, and demonstrated to be well-performed for complicated nonlinear equations.

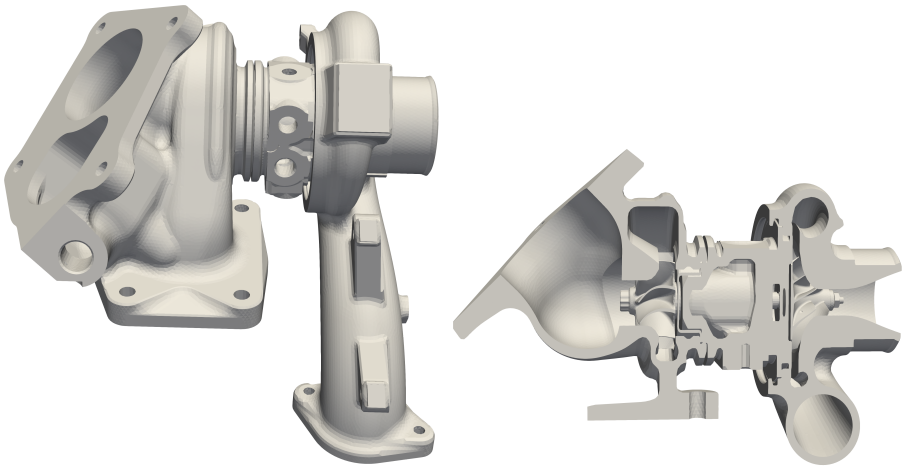
Chapter 8

A real–world engineering problem

The motivation of this research is to develop a robust iterative method to solve real–world engineering problems. In previous chapters, we have 'evolved' multigrid method via the local correction smoother, which recovers the smooth property in smoothing procedure of multigrid. It has been demonstrated that multigrid with local correction scheme has the capability of dealing with the contrived examples with low quality cells. We now turn to solve a real-world engineering problem to validate the effectiveness of the proposed local correction algorithm.

In this chapter, a thermomechanical simulation on a turbomachinery mesh is carried out. There is a commonly held view in turbomachinery industry that iterative solvers are not well-suitable for large-size thermomechanical simulations due to the poor robustness. We use a specific turbocharger mesh here to assert that a main factor that influences the convergence of iterative solvers is the mesh quality. The proposed local correction scheme provides an opportunity to tackle this barrier of the low quality mesh and improves the robustness of using multigrid. Our aim is to show that the convergence of multigrid to solve this real–world engineering problem can be significantly improved by the local correction smoother.

The problem chosen here is from the work [93], but simplified in terms of the boundary conditions and problem parameters. Algebraic multigrid is employed to solve both linear and nonlinear equations. The information of the turbocharger mesh is presented in section 8.1. Section 8.2 formulates the thermomechanical model as well as the problem setting. The implementation strategies are explained in section 8.3, especially the set-up of multigrid and local correction. Numerical results including the convergence rate are shown in section 8.4. Finally, we discuss some concluding remarks in section 8.5.



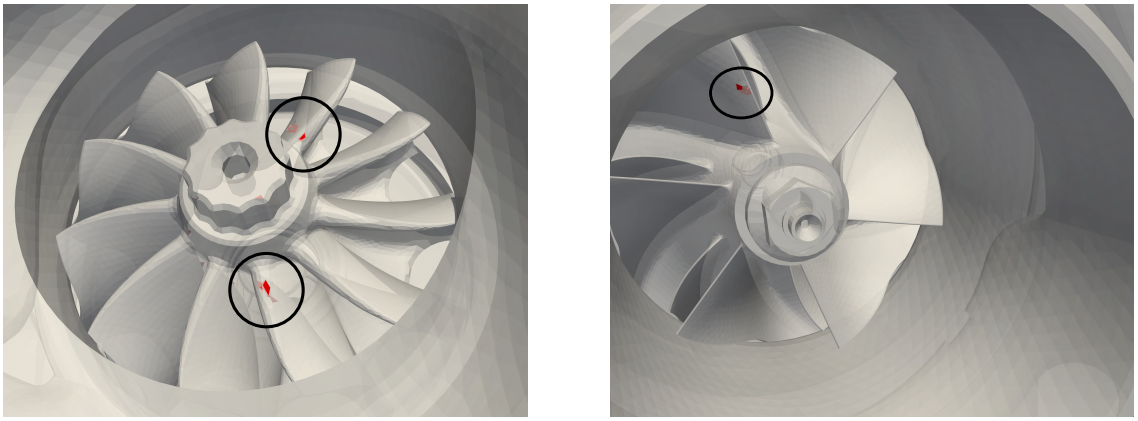
(a) Exterior view (b) Cut-away view

Fig. 8.1 Turbocharger geometry.

8.1 A turbocharger mesh

The geometry considered here is a turbocharger casing, which is shown in fig. 8.1. There are two turbines in the turbocharger geometry, which are located in the left and right hands of the domain. The mesh contains $1,603,438$ cells and $375,352$ vertices. When meshing the two turbines, several low quality cells are present. The low quality cells with normalised radius ratio smaller than 0.01, which are coloured red, in the two turbines are shown in fig. 8.2. It is found that several low quality cells are randomly located in both turbines. These low quality cells are not easy to fix locally because of the lack of the *a priori* information.

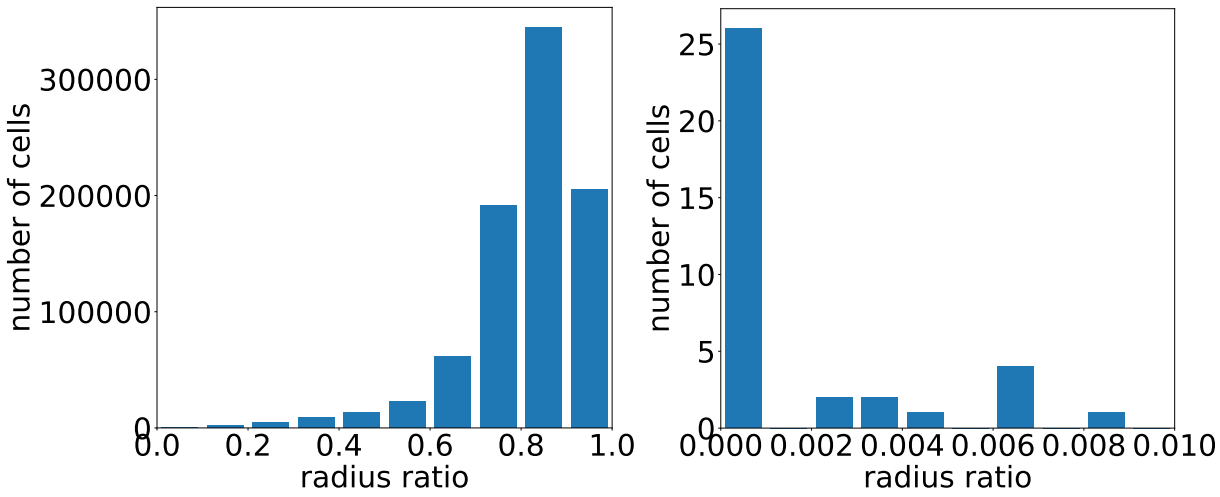
The histogram of the normalised radius ratio formulated in eq. (4.1) for this turbocharger mesh is shown in fig. 8.3. For this complicated engineering mesh, the radius ratio histogram is similar to the previous examples. It is still observed that most cells in this mesh are in high quality with the normalised radius ratio greater than 0.5. It is noted that there are a number of low quality cells with radius ratio smaller than 0.1, and these cells are located in many places of the domain. Now, we encounter a challenge for our local correction method that too many low quality cells make the local correction highly expensive. Even if the local correction smoother helps improve the convergence, the overall computational time would not be satisfactory. Thus, we need to reconsider the case and make some adjustments to our method. The local correction region for this problem is tracked by finding the cells of normalised radius ratio in $(0, 0.01)$, for which, we find, these cells are the ones inside the turbines. There are still a very small number of these low quality cells (around 50) in this mesh. Via numerical experiments, it is shown that these cells



(a) Low quality region on left turbine

(b) Low quality region on right turbine

Fig. 8.2 Low quality cells coloured red on the two turbines of the turbocharger mesh.



(a) radius ratio: $0-1$

(b) radius ratio: $0-0.01$

Fig. 8.3 Histogram of the normalised radius ratio of the turbocharger engine mesh.

8.2 Thermoelasticity problem

The thermomechanical model is based on a coupled system including a linear elasticity equation and a nonlinear heat equation. We consider the steady state solution to the general problem shown in [93]. The mechanical part of the coupled system is governed by the elasticity

$$-\operatorname{div}\sigma(u) = f \quad \text{in} \quad \Omega, \tag{8.1}$$

where u is the unknown displacement field, σ is the stress, and f is the body force. The stress is dependent on the strain and temperature field, given by

$$\sigma(u) = 2\mu\,(\varepsilon(u) - \varepsilon_T) + \lambda \operatorname{tr}(\varepsilon(u) - \varepsilon_T) I, \tag{8.2}$$

where μ and λ are the Lamé constants based on the material property, $\varepsilon(u)$ is the regular strain defined in eq. (4.24), and ε_T is the thermal strain given by

$$\varepsilon_T = (T - T_{\text{ref}})I, \tag{8.3}$$

in which T is the temperature field, and T_{ref} is the fixed reference temperature. We assume there is no body force, i.e. $f = (0,0,0)$. A uniform load of $(0,0,-10^3)$ is applied on the most bottom boundary of the domain. Other boundaries are taken as the free boundaries.

The temperature field T is governed by a nonlinear steady state heat equation

$$\begin{aligned} -\nabla \cdot (q(T)\nabla T) &= 0 \quad \text{in } \Omega, \\ T &= T_{\text{ref}} \quad \text{on } \partial\Omega, \end{aligned} \tag{8.4}$$

where $q(T)$ is a function of T given by

$$q(T) = 1 + T^2. \tag{8.5}$$

The Lamé constants are simplified here, which are set as temperature independent. The property of aluminium are used with Young's modulus 69 GPa and the Poisson's ratio 0.33. The reference temperature is chosen as the open air temperature 293.15K. The coupled equations are nonlinear due to the nonlinearity in the heat equation. It is noted that the coupling between the mechanical and thermal parts is one–way, namely, the elasticity depends on the temperature field but the steady state heat equation does not depend on the displacement field.

8.3 Solver strategy

Finite element method is carried out with linear (P_1) and quadratic (P_2) bases on simplices. Then a coupled algebraic system is obtained including a linear system of displacement field, and a nonlinear system of temperature field. We first solve the thermal problem to get the temperature field, followed by solving the elasticity problem using the input of the obtained temperature field. Therefore, this system solver is in the form of a block nonlinear Gauss–Seidel iterative process.

The nonlinear system generated by the nonlinear heat equation is solved by the Newton–multigrid method. For the scalar–valued heat equation problem, classical AMG is employed. Five classical AMG V-cycles are used in each Newton iteration to solve the Jacobian system. Symmetric Gauss–Seidel is taken as the global smoother for classical AMG, and applied twice for the pre- and post-smoothing in the classical AMG.

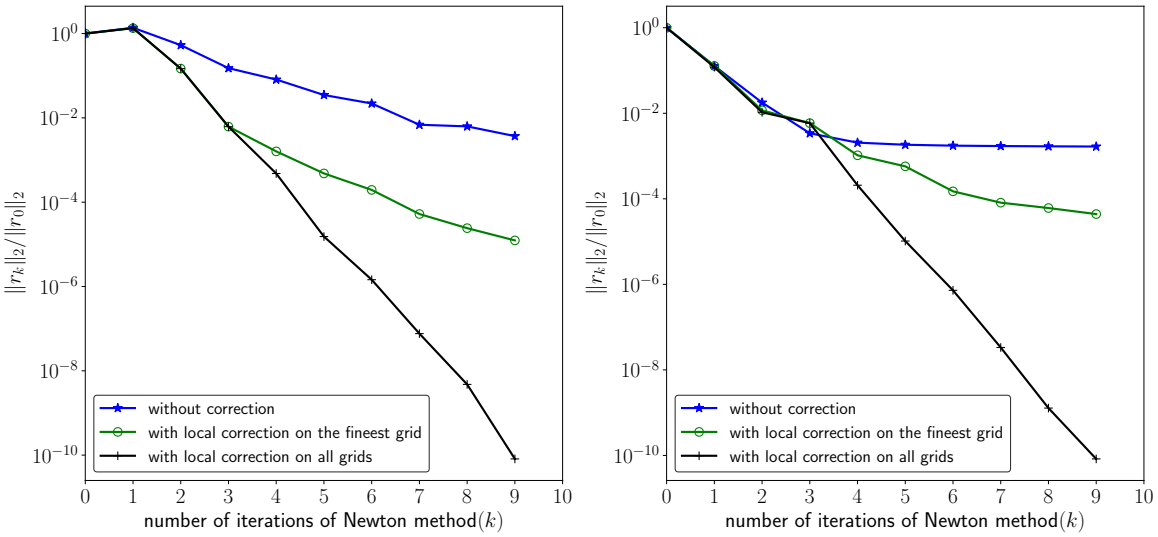
On the other hand, the linear system generated by the elasticity equation is solved by the multigrid preconditioned conjugate gradient method. For the vector-valued linear elasticity problem, smoothed aggregation AMG is served as the preconditioner. In each iteration of the conjugate gradient, two times of the preconditioner are applied. The global smoother for the smoothed aggregation AMG is chosen by the Jacobi preconditioned Chebyshev method. Two applications of the smoother are applied in terms of pre- and post-smoothing in the smoothed aggregation AMG.

The coarsening factor for both classical AMG (eq. (3.27) in section 3.6.1) and smoothed aggregation AMG (eq. (3.37) in section 3.6.2) is still taken as 0.08 for P_1 element and 0.05 for P_2 element. The local correction region in this problem is tracked by finding those cells with normalised radius ratio smaller than 0.01. Local correction is applied on the finest level in solving both linear and nonlinear systems. Local correction on the abstract coarse grids discussed in section 5.2 and the shifted eigenvalue technique introduced in section 6.2 are also served as parts of local correction in AMG. When using the smoothed aggregation AMG, local correction is only carried out on the first two levels. The Newton–multigrid solver for the nonlinear heat equation is terminated once the relative residual of the Newton's method reaches 10^{-10}, while we terminate the multigrid preconditioned conjugate gradient method for solving the elasticity equation when the relative residual of the conjugate gradient is smaller than 10^{-8}.

The implementation of AMG and local correction can be followed by the discussion in sections 5.3 and 6.3. Finite element simulations are constructed using libraries from the FEniCS Project [5, 80, 79]. Iterative solvers including Newton's methods, multigrid and conjugate gradient are implemented using the PETSc library [11, 10, 12].

Mesh level	1	2	3	4
Element type		P_1		
Number of DOFs in Ω	206995	48878	7580	1000
Number of DOFs in Ω_B	301	272	327	-
Element type		P_2		
Number of DOFs in Ω	1397533	128019	20026	4526
Number of DOFs in Ω_B	1416	402	362	-

Table 8.1 Problem sizes of the finite element and local correction systems on levels of using the classical AMG.



(a) P_1 element

(b) P_2 element

Fig. 8.4 Relative residual obtained by Newton-multigrid method with classical AMG used as the inner solver working on the nonlinear thermal problem.

8.4 Numerical results

Let us start with the results of the nonlinear heat equation. The problem size of the classical AMG and the local correction system on each level is shown in table 8.1. Clearly, the sizes of the local correction systems on different levels are kept similar, which still take up a very small portion of the entire problem size. In order to test the performance of the local correction on classical AMG discussed in chapter 5, we apply the local correction to (i) the finest grid only and (ii) the finest grid and the abstract coarse grids (except coarsest grid). The convergence rate of the relative residual computed by Newton–multigrid method with the classical AMG as the inner solver to solve the nonlinear hear equation eq. (8.4) is shown in fig. 8.4 for both P_1 and P_2 elements. Without any local correction, the Newton-

Element type	Level	1	2	3	4
P_1 element	Unshifted largest eigenvalue	7.819043	3.822354	2.543022	2.009753
	Shifted largest eigenvalue	4.09457	2.923119	2.367324	2.010555
P_2 element	Unshifted largest eigenvalue	8.845961	5.651259	2.468138	2.292369
	Shifted largest eigenvalue	6.217467	3.693972	2.453989	2.268228

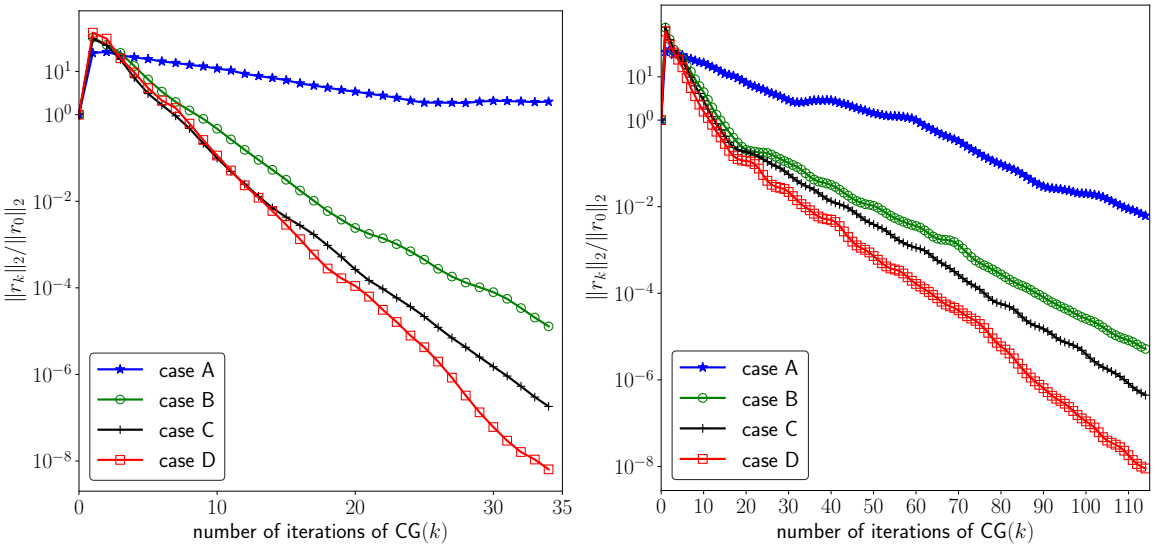
Table 8.2 Largest eigenvalues used in the smoothed prolongation of smoothed aggregation AMG to solve the thermal equation.

multigrid method stagnates at 10^{-3}. The local correction on the finest grid of the classical AMG improves the convergence rate of the Newton–multigrid method. If applying the local correction on both finest and abstract coarse grids, an improving convergence rate can be achieved. The local correction of classical AMG successfully recovers the poor convergence of the Newton–multigrid method. A relatively more accurate temperature field is obtained to input in the mechanical problem.

To test the local correction in smoothed aggregation AMG discussed in chapter 6, the smoothed prolongation operator in eq. (3.40) is constructed with (i) unshifted largest eigenvalue of the whole system, and (ii) shifted largest eigenvalue of the modified system in eq. (6.3). The largest eigenvalues used in this problem on different levels to construct the smoothed prolongation are listed in table 8.2. The unshifted largest eigenvalues of the whole system are generally larger than the shifted ones of the modified system, especially on the first two levels. There is no remarkable raise in largest eigenvalue on coarse levels, and local corrections are omitted on the levels three and four. It is also noticed that the increase in largest eigenvalues for this real-world problem is less significant compared to the numerical examples we make in section 6.3, which is because we lower the radius ratio threshold to track the low quality cells in this problem.

When applying the Jacobi preconditioned Chebyshev smoother, largest eigenvalues are also needed. As discussed in section 4.3.4, the same shifted largest eigenvalues can be used in construing the Chebyshev smoother. In order to have a full view of the convergence rate with respect to different largest eigenvalues used, the following four cases are considered:

- case A: without any local correction and unshifted largest eigenvalues used in both smoothed prolongation and Chebyshev smoother.
- case B: local correction with unshifted largest eigenvalues used in both smoothed prolongation and Chebyshev smoother.
- case C: local correction with shifted largest eigenvalues used in smoothed prolongation and unshifted largest eigenvalues used in the Chebyshev smoother.



(a) P_1 element (b) P_2 element

Fig. 8.5 Relative residual obtained by multigrid preconditioned conjugate gradient method with smoothed aggregation AMG as preconditioner working on the linear elasticity problem.

Case	Number of iterations needed for P_1 element	Number of iterations needed for P_2 element
Case A	>200	>1000
Case B	55	170
Case C	42	151
Case D	35	113

Table 8.3 Number of iteration counts for the smoothed aggregation AMG preconditioned conjugate gradient method to reach relative residual of 10^{-8} for solving the elasticity equation.

- case D: local correction with shifted largest eigenvalues used in both smoothed prolongation and Chebyshev smoother.

The convergence results of the four cases are shown in fig. 8.5 for both P_1 and P_2 elements. The obtained convergence rates are not surprising, similar to the previous examples we make. Without any local correction, the convergence of multigrid preconditioned conjugate gradient method completely stops. With the help of local correction, the solver converges faster within a limited number of iterations. And, the shifted largest eigenvalue technique can also accelerate the convergence rate.

Moreover, we list the detailed number of iterations of the multigrid preconditioned conjugate gradient needed for the relative residual to reach 10^{-8} in table 8.3. Clearly, it is much more difficult for the solver to converge with P_2 element, which has been commonly seen in engineering simulations and also observed in [93]. The local correction ensures the

Element type		P_1	P_2
FEM set–up		4.22	16.14
Multigrid set–up	Transfer operators	3.27	40.96
	Coarse grid systems	0.46	4.89
	Local correction system	0.56	0.68
Newton-multigrid iterations		1.21	14.28
Overall		9.72	76.95

Table 8.4 Computational time in seconds for relative residual to reach 10^{-10} solving the nonlinear heat equation via the Newton–multigrid method.

convergence of the solver. The largest eigenvalues in constructing the smoothed prolongation have a larger effect in convergence. The shifted largest eigenvalues used in smoothed prolongation can improve the convergence hugely by saving half of the computational time. On the other hand, the largest eigenvalues in Chebyshev smoother have a less effect. Shifted largest eigenvalue used in the Chebyshev smoother can save around 20% of computational costs.

Finally, the computational time is listed for both problems. We still cut the calculation procedure into three components: FEM set–up, multigrid set–up, and solver iterations. In particular, the computational time for multigrid set–up procedure is also split into three parts for the classical AMG: (i) construction of transfer operators, (ii) construction of coarse grid systems, (iii) construction of local correction systems including tracking low quality cells Ω_B, and there is another component for the smoothed aggregation AMG, which is (iv) finding the shifted largest eigenvalue $\lambda_{shifted}$. It is shown that tracking low quality cells and making local correction systems using FEniCs are extremely cheap for solving both problems and element types. It is noted that approximating the shifted largest eigenvalues is time–consuming, especially for P_2 element. In the multigrid set–up, the most time–consuming part is still the construction of the transfer operators. The computational time for solver iterations is much higher for P_2 element, especially for the elasticity problem since it takes 110 CG iterations for relative residual to reach 10^{-8}.

8.5 Concluding remarks

The presented example demonstrates two main discoveries in this research: (i) a main factor that the performance of multigrid degrades in solving engineering applications is the low cell quality of the mesh, and (ii) the local correction multigrid can recover the poor convergence of multigrid in presence of a small number of locally poor quality cells. We successfully solve this real-world turbomachinery problem by applying the proposed local correction

Element type		P_1	P_2
FEM set–up		12.27	76.27
Multigrid set–up	Transfer operators	22.18	278.14
	Coarse grid systems	6.38	77.50
	Local correction systems	1.56	2.01
	Finding $\lambda_{shifted}$	9.96	208.94
Newton-multigrid iterations		118.29	2055.56
Overall		170.64	2698.42

Table 8.5 Computational time in second for relative residual to reach 10^{-8} solving the linear elasticity equation by smoothed aggregation AMG preconditioned CG method.

scheme to tackle the issue of low quality meshes. In particular, both classical and smoothed aggregation AMG are carried out for both linear and nonlinear problems, which validates the performance of most techniques in this research including local correction on geometric grid level, local correction on the abstract coarse grid, and shifted eigenvalue technique. This work opens up the possibility of system-level turbomachinery modelling, and is strongly appealing for solving other complicated physical processes such as simulations of contact mechanics.

Chapter 9

Conclusions

In this thesis, we have considered using multigrid methods to deal with engineering meshes in presence of a small number of locally poor quality cells. It was observed that the performance of multigrid methods degrades significantly with low quality meshes. Numerical examples have been carried out to demonstrate that the poor convergence of multigrid is due to the local failure of the smoothing property of multigrid smoothers in regions of low quality cells. A global–local combined smoother has been developed for the geometric multigrid to deal with unstructured meshes with low quality cells, which includes two steps: a global smoother on the whole domain followed by a local correction on the subdomains with low quality cells. The combined smoother is effectively a Schwarz-type domain decomposition method with a full overlap and also in the form of a block Gauss–Seidel method.

The local correction smoother has been extended to the use of the algebraic multigrid (AMG) including the classical AMG and the smoothed aggregation AMG. It was found that parts of the high frequency error generated by the multigrid smoother in AMG propagates outward from areas of low quality cells on the fine grid. We suggested that these errors are generated on the abstract coarse grids. To apply the local correction properly on the abstract coarse grids of AMG, we developed an algorithm to track the low quality regions on the coarse grid via the information transfer between levels.

Low quality cells in a grid increase the spectral radius of the finite element system, which causes issues for those solvers that depend on the spectral radius. A shifted largest eigenvalue technique was proposed to approximate a reasonable value of the spectral radius with respect to the high quality regions in the grid. It has been employed in the construction of the Chebyshev smoother and the smoothed prolongation operator in the smoothed aggregation AMG.

Numerical experiments that range from illustrative cases to complicated applications have been largely carried out to validate the new smoother. Both linear and nonlinear

problems were tested on the unstructured and non-nested meshes. The poor convergence of multigrid for low quality meshes can be restored to the reference level with high quality meshes. In particular, we also applied the local correction method to solve a real-world thermomechanical simulation of turbomachinery problem. The performance demonstrated in this work opens up the possibility of the high–performance scalable multigrid methods to solve the complicated engineering applications at a system level.

We also give a summary of all the local correction techniques presented in this thesis in section 9.1. Some possible areas where this research can be further developed are listed in section 9.2.

9.1 Summary of local correction techniques

Local correction is the main technique discussed in this thesis. The idea is not only a smoother for multigrid, but also includes several schemes to apply the smoother appropriately in the case of geometric multigrid, algebraic multigrid, and Newton-multigrid methods. All the local correction schemes are illustrated in fig. 9.1.

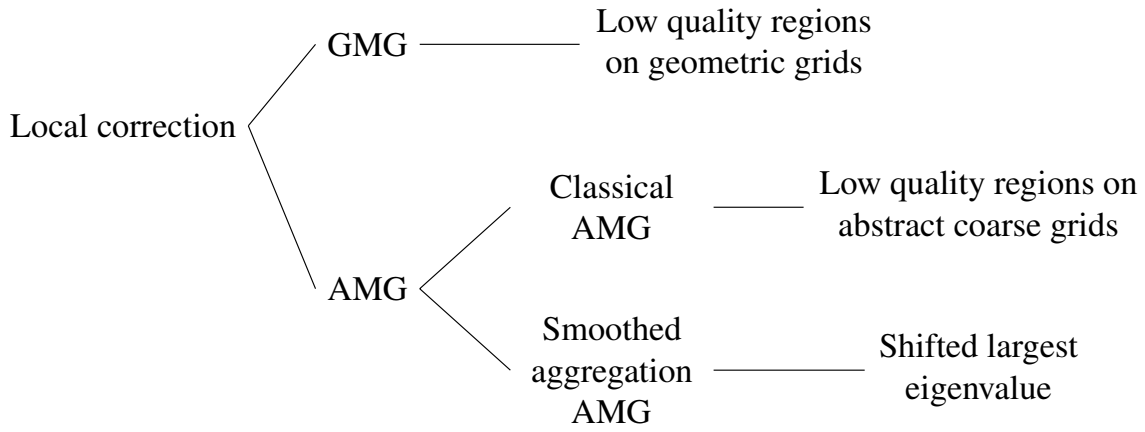


Fig. 9.1 A summary of local correction schemes in this thesis.

9.1.1 Local correction on geometric grids

To tackle the poor performance of the geometric multigrid (GMG) with low quality meshes, a global–local combined smoother is proposed. The smoother includes two steps: a global smoother on the whole domain to eliminate the high frequency error on most parts of domain, followed by a local correction smoother on local regions containing low quality cells to get rid of the remaining high frequency error. The local correction smoother is based on solving

the residual equation on local subdomains generated by extracting submatrix from the whole system (eq. (4.6)). In GMG, geometric mesh hierarchies are provided, the low quality region can be tracked by setting a threshold on the normalised radius ratio. The threshold used in this thesis is set as 0.1 for our contrived numerical examples and 0.01 for the turbomachinery problem.

9.1.2 Local correction on abstract grids of AMG

A highlight of this research is the extension of local correction to the algebraic multigrid (AMG) case. In AMG, coarse grids are provided algebraically, the geometric way to track low quality regions is not feasible. An algorithmic way to find the artificial low quality regions on the abstract coarse grid of AMG is developed via the information transfer between grid levels, which is explained in algorithm 5.1. Applying local correction on the low quality regions of abstract coarse grids can remove the remaining high frequency error produced on coarse grids.

9.1.3 Shifted largest eigenvalue

It is found that the presence of low quality cells increases the spectral radius of the linear system. If an algorithm depends on an estimate of the largest eigenvalue, then it would be inefficient in high quality regions due to the large spectral radius input. We propose a shifted largest eigenvalue strategy to decrease the value of spectral radius, which produces a reasonable input in the algorithm. There are two algorithms in this work that employs the shifted largest eigenvalues. The first is the use of the Chebyshev smoother in which if the unshifted largest eigenvalue is applied, the smoothing property is damaged globally. Another one is the construction of the smoothed prolongation in the smoothed aggregation AMG, in which if the spectral radius of the whole system is applied, then the energy of the coarse grid basis functions generated by the smoothed prolongation operator increases, leading to the inaccurate coarse grid correction in AMG.

9.2 Future work

The research presented in this thesis explores only a few potential avenues for the multigrid with local correction schemes. There are many other directions in which this research can be extended, including applications, implementations, and theories. We list three potential directions for future investigation.

9.2.1 More complicated applications

In this thesis, we focus on solving the elliptic type PDEs, mainly the Poisson equation and the linear elasticity. Actually, there are many more advanced problems which require much more careful treatments to solve. A natural question to ask is whether or not the proposed multigrid with local correction scheme can solve these advanced problems. Three problems are listed here which are of particular interest in applying the local correction idea.

- A next step of the work in chapter 8 could be the construction of a robust solver for the contact mechanics on the turbocharger geometry. Multigrid has been developed to deal with the general contact mechanics problems [4, 115]. These approaches are typically based on simple geometries and structured grids. A specific target in this area is to solve the contact problems on large complicated domains. The multigrid with local correction approach is capable of dealing with such meshes, so that it can be applicable to the complicated contact mechanics simulations in engineering.

- There are many low quality cells appearing in the large deformation problems [14, 73], in which the large deformation refers to that some cells in the mesh deform significantly during the dynamical process. Iterative solvers are not robust in solving the large deformation problems due to the presence of low quality cells [60, 62]. Then the local correction smoother is appealing to make a robust solver to solve large deformation problems.

- Rather than solving more complicated geometries, we may consider solving some more complicated equations. The variable-coefficient Poisson equation varies the coefficients in the Poisson equation in different subdomains. There is no doubt that if using a standard iterative solver to solve the variable-coefficient Poisson equation with some local varying coefficients, then the solver would fail locally, which is similar to the failure of the smoother in local regions of low quality cells. Some domain decomposition type methods have been created to solve this type of problems, such as [119, 61]. It is anticipated that multigrid with local correction can be served as a potential solver to solve the local variable-coefficient Poisson equation.

9.2.2 Mathematical underpinnings

Via numerical examples, we found that the low quality cells also have significant impact on the finite element solution error. Local correction can help reduce the error, but is unable to fix the error to the high quality mesh level. It is important to understand how the mesh quality influences the approximation property of the finite element method. We have discussed some recent research under this direction in section 4.1. However, there is still no direct clue to

make an explicit bound in the finite element error with respect to the cell quality. A good target is to find a threshold on the mesh quality which can ensure the accuracy of the finite element solution.

Another topic we are interested in is the local Fourier analysis in analyzing the smoothing property of the smoother. With a low quality mesh, it is found that the smoothing property is damaged locally in regions of low quality cells. Carrying out the rigorous local Fourier analysis on the low quality mesh is the best way to understand this phenomenon thoroughly. In the work of [82], the local Fourier analysis has been extended to the overlapping smoothers, which is of particular interest.

9.2.3 High–level implementation

The implementation of multigrid and local correction can still be improved. Many studies have been carried out on the parallel implementation of multigrid, see [25, 15]. Local correction is one of the domain decomposition methods, which can be easily implemented in parallel. An interesting topic is the matrix-free multigrid strategy [85, 81], which allows the scalable computation of engineering simulations by multigrid.

Here we propose a parallel implementation strategy for the local correction combined smoother. It is noted that many multigrid smoothers can be implemented in parallel, in particular for the two methods we use, Gauss–Seidel [3] and Chebyshev methods [2]. The key problem for us becomes how to implement the global–local combined smoother in parallel. First, the construction of local correction system is easily implemented in parallel. We can cut the whole domain into many pieces, and track low quality cells on each of piece on one processor. It is also easy to parallel implement the local correction by solving each local system on one processor. A proposed parallel implementation strategy for the combined smoother is given in the following.

- Parallel implementation of smoother on high quality region Ω_G and local correction on Ω_B
- Residual correction on Ω_B, and then applying smoother on Ω_B
- Parallel implementation of local correction on Ω_B

On the other hand, it is noted in chapter 6 that the local correction for the smoothed aggregation AMG requires to reconstruct the smoothed prolongation eq. (3.40), which is found to be time–consuming. A parallel implementation of this step is also necessary. A proposed approach is given here.

- At the same time, on different processors carry out
  - Find the tentative prolongation $\tilde{P}_l$
  - Calculate the matrix multiplication $D^{-1}A$
  - Approximate the shifted largest eigenvalue $\lambda_{\max}$
- Calculate the smoothed prolongation by combining all the pieces obtained from previous step

With the proposed parallel implementation strategy, our multigrid with local correction scheme can be accelerated. In this thesis, the largest problem we can solve is in million–size, i.e. $O(10^6)$. An important target is to enable multigrid with local correction to deal with billion–size $O(10^9)$ complicated problem.

References

[1] Adams, M., Brezina, M., Hu, J., and Tuminaro, R. (2003a). Parallel multigrid smoothing: polynomial versus Gauss–Seidel. *Journal of Computational Physics*, 188(2):593–610.

[2] Adams, M., Brezina, M., Hu, J., and Tuminaro, R. (2003b). Parallel multigrid smoothing: polynomial versus Gauss—Seidel. *Journal of Computational Physics*, 188(2):593–610.

[3] Adams, M. F. (2001). A distributed memory unstructured Gauss-Seidel algorithm for multigrid smoothers. In *SC '01: Proceedings of the 2001 ACM/IEEE Conference on Supercomputing*, pages 14–14.

[4] Adams, M. F. (2004). Algebraic multigrid methods for constrained linear systems with applications to contact problems in solid mechanics. *Numerical Linear Algebra with Applications*, 11(2):141–153.

[5] Alnæs, M. S., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M. E., and Wells, G. N. (2015). The FEniCS Project Version 1.5. *Archive of Numerical Software*, 3(100):9–23.

[6] Amodio, P. and Mazzia, F. (1995). A parallel Gauss–Seidel method for block tridiagonal linear systems. *SIAM Journal on Scientific Computing*, 16(6):1451–1461.

[7] Arnoldi, W. E. (1951). The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quarterly of Applied Mathematics*, 9(1):17–29.

[8] Ashby, S. F. and Falgout, R. D. (1996). A parallel multigrid preconditioned conjugate gradient algorithm for groundwater flow simulations. *Nuclear Science and Engineering*, 124(1):145–159.

[9] Babuška, I. and Aziz, A. K. (1976). On the angle condition in the finite element method. *SIAM Journal on Numerical Analysis*, 13(2):214–226.

[10] Balay, S., Abhyankar, S., Adams, M. F., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W. D., Kaushik, D., Knepley, M. G., May, D. A., McInnes, L. C., Mills, R. T., Munson, T., Rupp, K., Sanan, P., Smith, B. F., Zampini, S., Zhang, H., and Zhang, H. (2018a). PETSc users manual. Technical Report ANL-95/11 - Revision 3.10, Argonne National Laboratory.

[11] Balay, S., Abhyankar, S., Adams, M. F., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W. D., Kaushik, D., Knepley, M. G., May, D. A., McInnes, L. C., Mills, R. T., Munson, T., Rupp, K., Sanan, P., Smith, B. F., Zampini, S., Zhang, H., and Zhang, H. (2018b). PETSc Web page.

[12] Balay, S., Gropp, W. D., McInnes, L. C., and Smith, B. F. (1997). Efficient management of parallelism in object oriented numerical software libraries. In Arge, E., Bruaset, A. M., and Langtangen, H. P., editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press.

[13] Bank, R. E. and Dupont, T. (1981). An optimal order process for solving finite element equations. *Mathematics of Computation*, 36(153):35–51.

[14] Bathe, K.-J., Ramm, E., and Wilson, E. L. (1975). Finite element formulations for large deformation dynamic analysis. *International Journal for Numerical Methods in Engineering*, 9(2):353–386.

[15] Bergen, B., Gradl, T., Hulsemann, F., and Rude, U. (2006). A massively parallel multigrid method for finite elements. *Computing in Science & Engineering*, 8(6):56–62.

[16] Bittencourt, M. L., Douglas, C. C., and Feijóo, R. A. (2001). Nonnested multigrid methods for linear problems. *Numerical Methods for Partial Differential Equations*, 17(4):313–331.

[17] Bochev, P. B., Garasi, C. J., Hu, J. J., Robinson, A. C., and Tuminaro, R. S. (2003). An improved algebraic multigrid method for solving Maxwell's equations. *SIAM Journal on Scientific Computing*, 25(2):623–642.

[18] Brabazon, K. J., Hubbard, M. E., and Jimack, P. K. (2014). Nonlinear multigrid methods for second order differential operators with nonlinear diffusion coefficient. *Computers & Mathematics with Applications*, 68(12):1619–1634.

[19] Braess, D. (2007). *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*. Cambridge University Press.

[20] Braess, D. and Hackbusch, W. (1983). A new convergence proof for the multigrid method including the V-cycle. *SIAM Journal on Numerical Analysis*, 20(5):967–975.

[21] Bramble, J. H., Ewing, R. E., Pasciak, J. E., and Schatz, A. H. (1988). A preconditioning technique for the efficient solution of problems with local grid refinement. *Computer Methods in Applied Mechanics and Engineering*, 67(2):149–159.

[22] Bramble, J. H. and Zhang, X. (2000). The analysis of multigrid methods. *Handbook of Numerical Analysis*, 7:173–415.

[23] Brandt, A. (1973). Multi-level adaptive technique (MLAT) for fast numerical solution to boundary value problems. In *Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics*, pages 82–89. Springer.

[24] Brandt, A. (1977). Multi–level adaptive solutions to boundary–value problems. *Mathematics of Computation*, 31(138):333–390.

[25] Brandt, A. (1981). Multigrid solvers on parallel computers. In *Elliptic Problem Solvers*, pages 39–83. Elsevier.

[26] Brenner, S. and Scott, R. (2007). *The Mathematical Theory of Finite Element Methods*, volume 15. Springer Science & Business Media.

[27] Briggs, W. L., Henson, V. E., and McCormick, S. F. (2000). *A Multigrid Tutorial*. SIAM, second edition.

[28] Brune, P. R., Knepley, M. G., Smith, B. F., and Tu, X. (2015). Composing scalable nonlinear algebraic solvers. *SIAM Review*, 57(4):535–565.

[29] Chen, Y. (2019). Supporting material. URL https://github.com/yc397/multigrid_examples.

[30] Ciarlet, P. G. (2002). *The Finite Element Method for Elliptic Problems*. SIAM.

[31] Dembo, R. S., Eisenstat, S. C., and Steihaug, T. (1982). Inexact newton methods. *SIAM Journal on Numerical Analysis*, 19(2):400–408.

[32] Deuflhard, P. (2011). *Newton Methods for Nonlinear Problems: Affine Invariance and Adaptive Algorithms*, volume 35. Springer Science & Business Media.

[33] Dohrmann, C. R. (2003). A preconditioner for substructuring based on constrained energy minimization. *SIAM Journal on Scientific Computing*, 25(1):246–258.

[34] Dolean, V., Jolivet, P., and Nataf, F. (2015). *An Introduction to Domain Decomposition Methods: Algorithms, Theory, and Parallel Implementation*, volume 144. SIAM.

[35] Du, Q., Wang, D., and Zhu, L. (2009). On mesh geometry and stiffness matrix conditioning for general finite element spaces. *SIAM Journal on Numerical Analysis*, 47(2):1421–1444.

[36] Duprez, M., Lleras, V., and Lozinski, A. (2019). Finite element method with local damage of the mesh. *ESAIM: M2AN*, 53(6):1871–1891.

[37] Eisenstat, S. C. and Walker, H. F. (1996). Choosing the forcing terms in an inexact Newton method. *SIAM Journal on Scientific Computing*, 17(1):16–32.

[38] Evangelista Jr, F., de Sousa Alves, G., Moreira, J. F. A., and de Paiva, G. O. F. (2020). A global–local strategy with the generalized finite element framework for continuum damage models. *Computer Methods in Applied Mechanics and Engineering*, 363:112888.

[39] Farhat, C., Lesoinne, M., LeTallec, P., Pierson, K., and Rixen, D. (2001). FETI-DP: a dual–primal unified FETI method—part I: A faster alternative to the two-level FETI method. *International Journal for Numerical Methods in Engineering*, 50(7):1523–1544.

[40] Farhat, C. and Roux, F.-X. (1991). A method of finite element tearing and interconnecting and its parallel solution algorithm. *International Journal for Numerical Methods in Engineering*, 32(6):1205–1227.

[41] Fedorenko, R. P. (1962). A relaxation method for solving elliptic difference equations. *USSR Computational Mathematics and Mathematical Physics*, 1(4):1092–1096.

[42] Fedorenko, R. P. (1964). The speed of convergence of one iterative process. *USSR Computational Mathematics and Mathematical Physics*, 4(3):227–235.

[43] Fragakis, Y. and Papadrakakis, M. (2003). The mosaic of high performance domain decomposition methods for structural mechanics: Formulation, interrelation and numerical efficiency of primal and dual methods. *Computer Methods in Applied Mechanics and Engineering*, 192(35):3799–3830.

[44] Franceschini, A., Magri, V. A. P., Mazzucco, G., Spiezia, N., and Janna, C. (2019). A robust adaptive algebraic multigrid linear solver for structural mechanics. *Computer Methods in Applied Mechanics and Engineering*, 352:389–416.

[45] Freitag, L. A. and Ollivier-Gooch, C. (2000). A cost/benefit analysis of simplicial mesh improvement techniques as measured by solution efficiency. *International Journal of Computational Geometry & Applications*, 10(4):361–382.

[46] Gee, M. W., Hu, J. J., and Tuminaro, R. S. (2009). A new smoothed aggregation multigrid method for anisotropic problems. *Numerical Linear Algebra with Applications*, 16(1):19–37.

[47] George, A. (1973). Nested dissection of a regular finite element mesh. *SIAM Journal on Numerical Analysis*, 10(2):345–363.

[48] Geuzaine, C. and Remacle, J.-F. (2009). Gmsh: A three-dimensional finite element mesh generator with built-in pre-and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331.

[49] Griebel, M. and Oswald, P. (1995). On the abstract theory of additive and multiplicative schwarz algorithms. *Numerische Mathematik*, 70(2):163–180.

[50] Guillard, H., Janka, A., and Vaněk, P. (2008). Analysis of an algebraic Petrov–Galerkin smoothed aggregation multigrid method. *Applied Numerical Mathematics*, 58(12):1861–1874.

[51] Gutknecht, M. H. and Röllin, S. (2002). The Chebyshev iteration revisited. *Parallel Computing*, 28(2):263–283.

[52] Hackbusch, W. (1978). A fast iterative method for solving Poisson's equation in a general region. In *Numerical Treatment of Differential Equations*, pages 51–62. Springer.

[53] Hackbusch, W. (1992). Comparison of different multi-grid variants for nonlinear equations. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 72(2):148–151.

[54] Hackbusch, W. (2013). *Multi-grid Methods and Applications*, volume 4. Springer Science & Business Media.

[55] Haftka, R. T. (1990). Stiffness–matrix condition number and shape sensitivity errors. *AIAA Journal*, 28(7):1322–1324.

[56] Hageman, L. A. and Young, D. M. (2012). *Applied Iterative Methods*. Courier Corporation.

[57] Hannukainen, A., Korotov, S., and Křížek, M. (2012). The maximum angle condition is not necessary for convergence of the finite element method. *Numerische Mathematik*, 120(1):79–88.

[58] Hemker, P. (1990). On the order of prolongations and restrictions in multigrid procedures. *Journal of Computational and Applied Mathematics*, 32(3):423–429.

[59] Hernández, V., Román, J. E., Tomás, A., and Vidal, V. (2007). Krylov–Schur methods in SLEPc. *Universitat Politecnica de Valencia, Tech. Rep. STR-7*.

[60] Hesch, C. and Betsch, P. (2010). Transient three-dimensional domain decomposition problems: Frame–indifferent mortar constraints and conserving integration. *International Journal for Numerical Methods in Engineering*, 82(3):329–358.

[61] Hou, S. and Liu, X.-D. (2005). A numerical method for solving variable coefficient elliptic equation with interfaces. *Journal of Computational Physics*, 202(2):411–445.

[62] Huang, J., Liu, X., Bao, H., Guo, B., and Shum, H.-Y. (2006). An efficient large deformation method using domain decomposition. *Computers & Graphics*, 30(6):927–935.

[63] Iwamura, C., Costa, F. S., Sbarski, I., Easton, A., and Li, N. (2003). An efficient algebraic multigrid preconditioned conjugate gradient solver. *Computer Methods in Applied Mechanics and Engineering*, 192(20):2299–2318.

[64] Ji, Z., Fu, L., Hu, X., and Adams, N. (2020). A consistent parallel isotropic unstructured mesh generation method based on multi-phase SPH. *Computer Methods in Applied Mechanics and Engineering*, 363:112881.

[65] Katz, A. and Sankaran, V. (2011). Mesh quality effects on the accuracy of CFD solutions on unstructured meshes. *Journal of Computational Physics*, 230(20):7670–7686.

[66] Kelley, C. T. (2003). *Solving Nonlinear Equations with Newton's Method*, volume 1. SIAM.

[67] Klingner, B. M. and Shewchuk, J. R. (2008). Aggressive tetrahedral mesh improvement. In *Proceedings of the 16th International Meshing Roundtable*, pages 3–23. Springer, Berlin, Heidelberg.

[68] Knupp, P. M. (2000a). Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities. Part I – a framework for surface mesh optimization. *International Journal for Numerical Methods in Engineering*, 48(3):401–420.

[69] Knupp, P. M. (2000b). Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantities. part II - a framework for volume mesh optimization and the condition number of the jacobian matrix. *International Journal for Numerical Methods in Engineering*, 48(8):1165–1185.

[70] Knupp, P. M. (2001). Algebraic mesh quality metrics. *SIAM Journal on Scientific Computing*, 23(1):193–218.

[71] Křížek, M. (1992). On the maximum angle condition for linear tetrahedral elements. *SIAM Journal on Numerical Analysis*, 29(2):513–520.

[72] Lanczos, C. (1950). An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of Research of the National Bureau of Standards*, 45(4):255–282.

[73] Laursen, T. and Simo, J. C. (1993). A continuum–based finite element formulation for the implicit solution of multibody, large deformation–frictional contact problems. *International Journal for Numerical Methods in Engineering*, 36(20):3451–3485.

[74] Li, J. and Widlund, O. B. (2006). FETI-DP, BDDC, and block Cholesky methods. *International Journal for Numerical Methods in Engineering*, 66(2):250–271.

[75] Lions, P.-L. (1988). On the Schwarz alternating method. I. In *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, volume 1, page 42. Paris, France.

[76] Lions, P.-L. (1990). On the Schwarz alternating method. III: a variant for nonoverlapping subdomains. In *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, volume 6, pages 202–223. SIAM Philadelphia, PA.

[77] Llorente, I. M., Diskin, B., and Melson, N. D. (2000). Alternating plane smoothers for multiblock grids. *SIAM Journal on Scientific Computing*, 22(1):218–242.

[78] Llorente, I. M. and Melson, N. D. (2000). Behavior of plane relaxation methods as multigrid smoothers. *Electronic Transactions on Numerical Analysis*, 10:92–114.

[79] Logg, A., Mardal, K.-A., and Wells, G. N. (2012). *Automated solution of differential equations by the finite element method: The FEniCS book*, volume 84. Springer Science & Business Media.

[80] Logg, A. and Wells, G. N. (2010). DOLFIN: Automated finite element computing. *ACM Transactions on Mathematical Software*, 37(2):20.

[81] Lv, X., Zhao, Y., Huang, X., Xia, G., and Su, X. (2007). A matrix–free implicit unstructured multigrid finite volume method for simulating structural dynamics and fluid–structure interaction. *Journal of Computational Physics*, 225(1):120–144.

[82] MacLachlan, S. P. and Oosterlee, C. W. (2011). Local fourier analysis for multigrid with overlapping smoothers applied to systems of PDEs. *Numerical Linear Algebra with Applications*, 18(4):751–774.

[83] Mandel, J., Dohrmann, C. R., and Tezaur, R. (2005). An algebraic theory for primal and dual substructuring methods by constraints. *Applied Numerical Mathematics*, 54(2):167–193.

[84] Mandel, J. and Tezaur, R. (2001). On the convergence of a dual-primal substructuring method. *Numerische Mathematik*, 88(3):543–558.

[85] May, D. A., Brown, J., and Le Pourhiet, L. (2015). A scalable, matrix–free multigrid preconditioner for finite element discretizations of heterogeneous stokes flow. *Computer Methods in Applied Mechanics and Engineering*, 290:496–523.

[86] Nicolaides, R. A. (1977). On the l^2 convergence of an algorithm for solving finite element equations. *Mathematics of Computation*, 31(140):892–906.

[87] Olshanskii, M. A. and Tyrtshnikov, E. E. (2014). *Iterative Methods for Linear Systems: Theory and Applications*. SIAM.

[88] Oosterlee, C. W. (1997). A GMRES-based plane smoother in multigrid to solve 3D anisotropic fluid flow problems. *Journal of Computational Physics*, 130(1):41–53.

[89] Parthasarathy, V. N., Graichen, C. M., and Hathaway, A. F. (1994). A comparison of tetrahedron quality measures. *Finite Elements in Analysis and Design*, 15(3):255–261.

[90] Poulson, J. L. (2012). *Fast parallel solution of heterogeneous 3D time–harmonic wave equations*. PhD thesis, University of Texas at Austin.

[91] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical Recipes: The Art of Scientific Computing*. Cambridge university press, third edition.

[92] Reusken, A. (1987). Convergence of the multigrid full approximation scheme for a class of elliptic mildly nonlinear boundary value problems. *Numerische Mathematik*, 52(3):251–277.

[93] Richardson, C. N., Sime, N., and Wells, G. N. (2019). Scalable computation of thermomechanical turbomachinery problems. *Finite Elements in Analysis and Design*, 155:32–42.

[94] Ruge, J. W. and Stüben, K. (1987). Algebraic multigrid. In *Multigrid Methods*, pages 73–130. SIAM.

[95] Saad, Y. (2003). *Iterative Methods for Sparse Linear Systems*. SIAM, second edition.

[96] Saad, Y. and Schultz, M. H. (1986). GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical computing*, 7(3):856–869.

[97] Sampath, R. S. and Biros, G. (2010). A parallel geometric multigrid method for finite elements on octree meshes. *SIAM Journal on Scientific Computing*, 32(3):1361–1392.

[98] Scott, L. R. and Zhang, S. (1992). Higher–dimensional non–nested multigrid methods. *Mathematics of Computation*, 58(198):457–466.

[99] Shadid, J. N., Tuminaro, R. S., and Walker, H. F. (1997). An inexact Newton method for fully coupled solution of the Navier–Stokes equations with heat and mass transport. *Journal of Computational Physics*, 137(1):155–185.

[100] Shaidurov, V. V. (2013). *Multigrid Methods for Finite Elements*. Springer Science & Business Media.

[101] Shang, Y. (2009). A distributed memory parallel Gauss–Seidel algorithm for linear algebraic systems. *Computers & Mathematics with Applications*, 57(8):1369–1376.

[102] Shewchuk, J. (2002). What is a good linear finite element? interpolation, conditioning, anisotropy, and quality measures. In *Proceedings of the 11th International Meshing Roundtable*, pages 115–126.

[103] Smith, B., Bjorstad, P., Gropp, W. D., and Gropp, W. (2004). *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press.

[104] Stewart, G. W. (2002). A Krylov–Schur algorithm for large eigenproblems. *SIAM Journal on Matrix Analysis and Applications*, 23(3):601–614.

[105] Stüben, K. (2001). A review of algebraic multigrid. In *Numerical Analysis: Historical Developments in the 20th Century*, pages 331–359. Elsevier.

[106] Thomas Dickopf (2010). *On Multilevel Methods Based on Non-Nested Meshes*. PhD thesis, Rheinische Friedrich-Wilhelms-Universität Bonn.

[107] Trefethen, L. N. and Bau III, D. (1997). *Numerical Linear Algebra*. SIAM.

[108] Trottenberg, U., Oosterlee, C. W., and Schuller, A. (2000). *Multigrid*. Academic Press, London.

[109] Vaněk, P., Brezina, M., and Mandel, J. (2001). Convergence of algebraic multigrid based on smoothed aggregation. *Numerische Mathematik*, 88(3):559–579.

[110] Vaněk, P., Mandel, J., and Brezina, M. (1996). Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, 56(3):179–196.

[111] Varga, R. S. (2000). *Matrix Iterative Analysis*. Springer.

[112] Wathen, A. J. (2015). Preconditioning. *Acta Numerica*, 24:329–376.

[113] Wesseling, P. and Oosterlee, C. W. (2001). Geometric multigrid with applications to computational fluid dynamics. *Journal of Computational and Applied Mathematics*, 128(1):311–334.

[114] Wienands, R. and Joppich, W. (2004). *Practical Fourier Analysis for Multigrid Methods*. Chapman and Hall/CRC.

[115] Wohlmuth, B. I. and Krause, R. H. (2003). Monotone multigrid methods on non-matching grids for nonlinear multibody contact problems. *SIAM Journal on Scientific Computing*, 25(1):324–347.

[116] Xu, J. (1992). Iterative methods by space decomposition and subspace correction. *SIAM Review*, 34(4):581–613.

[117] Xu, J. (2001). The method of subspace corrections. *Journal of Computational and Applied Mathematics*, 128(1):335–362.

[118] Xu, J. and Zikatanov, L. (2017). Algebraic multigrid methods. *Acta Numerica*, 26:591–721.

[119] Yang, K., Pouransari, H., and Darve, E. (2019). Sparse hierarchical solvers with guaranteed convergence. *International Journal for Numerical Methods in Engineering*, 120(8):964–986.

[120] Young, D. M. (2014). *Iterative Solution of Large Linear Systems*. Elsevier.

[121] Zhang, Y., Hughes, T. J. R., and Bajaj, C. L. (2010). An automatic 3D mesh generation method for domains with multiple materials. *Computer Methods in Applied Mechanics and Engineering*, 199(5):405–415.

[122] Zhou, Y. and Saad, Y. (2008). Block Krylov–Schur method for large symmetric eigenvalue problems. *Numerical Algorithms*, 47(4):341–359.